7th International Conference on Metaheuristics and Nature Inspired Computing
Oct. 27th-31 2018
Marrakech, Morocco

Proceedings of META'2018

7th International Conference on Metaheuristics and Nature Inspired computing

Sponsors

Inria informatiques mathématiques

cnrs

Université de Lille

EMI

gdr macs

CRIStAL Centre de Recherche en Informatique, Signal et Automatique de Lille

SYNERGY
Horizon 2020
GA No 692286

GdR RO CNRS – GdR 3002

LIRIMA

Table of contents

Unsupervised aerial image classification using fly algorithm and the Parisian approach, Maiza Mohammed [et al.] 1

Maturation of Individuals in Evolutionary Learning, Pavone Mario F. [et al.] 6

Multi Objective Optimization and Bionic Optimization Strategies in Engineering Design, Steinbuch Rolf 16

A Cuckoo Search Algorithm for the Flexible Ligand-Protein Docking problem, Khensous Ghania [et al.] 19

A Parallel Primal Heuristic to solve a Integrated Production Planning and Scheduling Problem, Menezes Gustavo [et al.] 22

WPBO: A New Metaheuristic Technique Inspired from Wolf Pack Behaviour, Toumi Abida [et al.] 31

A hybrid meta-heuristic-based multi-agent for Industry 4.0, Tetouani Samir 42

A constructive heuristic for two machine job shop scheduling problem under availability constraints on one machine, Benttaleb Mourad [et al.] 50

A heuristic rule for ranking scientific journals based on citation impact, Lando Tommaso [et al.] 53

Assessing Film Coefficients of Microchannel Heat Sinks, Cruz-Duarte Jorge Mario [et al.] 54

Advanced portfolio optimization problems, Tichy Tomas 62

A Matheuristic for the Rainbow Cycle Cover problem, Moreno Jorge [et al.] 63

Solving Multiple Sequence Alignment Problem Using a Discrete Hybrid Particle Swarm Optimization Algorithm., Deschênes Hugo [et al.] 70

Nature-inspired Deployment for Context-aware Services, Satoh Ichiro 73

New hybrid differential evolution algorithm for multiple objective optimization, Gagné Caroline [et al.] 83

On the Sensitivity of Grid-Based Parameter Adaptation Method, Tatsis Vasileios [et al.] 86

Application of fuzzy smoothing filter in empirical copula function, Kresta Ales 95

Machine-learning algorithms for portfolio optimization problems, Kouaissah Noureddine [et al.] 98

An Innovative Heuristic Mixed-Integer Optimization Approach for Multi-Criteria Optimization based Production Planning in the context of Production Smoothing, Kamhuber Felix [et al.] 101

New Approach for Continuous and Discrete Optimization: Optimization by Morphological Filters, Khelifa Chahinez [et al.] 110

A Combined Data Mining and Tabu Search approach for Single Customer Dial-a-Ride Problem, Morais Ana Catarina [et al.] 121

Fitting epidemiological models' parameters via multi-objective optimization, Ruiz Ferrández Miriam [et al.] 124

Iterated-Greedy-Based Metaheuristic with Tabu Search and Simulated Annealing for Solving Permutation Flow Shop Problem, Mesmar Khadija [et al.] 127

On VNS-GRASP and Iterated Greedy Metaheuristics for Solving Hybrid Flow Shop Scheduling Problem with Uniform Parallel Machines and Sequence Independent Setup Time, Aqil Said [et al.] 133

Dual tree wavelet transform based denoising of images using subband adaptive thresholding via genetic algorithm, Boukhobza Abdelkader [et al.] 148

A Bi-Objective Maintenance-Routing Problem; an efficient solving approach, Rahimi Mohammad [et al.] 155

Heat exchanger network synthesis with an enhanced superstructure and hybrid metaheuristics, Pavão Leandro [et al.] 158

Transformer's Health Index using Computational Intelligence, Alves Dos Santos Ramon [et al.] 165

Application of the surrogate models for protein structure prediction, Rakhshani Hojjat [et al.] 175

A Hybrid Genetic Algorithm for the job shop problem with transportation and blocking no wait constraints, Louaqad Saad [et al.] 178

A randomized search procedure combined with simulated annealing for the capacitated location routing problem, Ali Lemouari 188

Design and Parallel Implementation of the H264 application on Heterogeneous Architectures, Adda Chahrazed [et al.] 191

A Parallel Adaptive Differential Evolution Algorithm for Electric Motor Design, Essaid Mokhtar [et al.] 204

Virtual screening in electrostatic potential using an evolutionary algorithm, Puertas-Martín S. [et al.] 207

Improved NSGAII Based on a Multiple-Criteria Decision Analysis Method for Business Process Optimization, Mahammed Nadir [et al.] 210

Efficient Generic Support for Global Routing Constraints in Constraint-Based

Local Search Frameworks, Meurisse Quentin [et al.] 221

A New Hidden Markov Model Approach for Pheromone Level Exponent Adaptation in Ant Colony System, Bouzbita Safae 232

An algorithm based on dimensionality reduction through parameterized curves to solve a class of non-convex global optimization, Mohamed Rahal 241

Metaheuristics for Agent based Intelligent Evacuation System, Hajjem Manel [et al.] 244

Vector-Quantization Codebook Generation using LBG and Meta-Heuristic Algorithms, Boubechal Ikram [et al.] 247

Multi-gene genetic programming for feature selection in DNA Microarrays, Sfaksi Sara [et al.] 250

A modified cuckoo search algorithm for unsupervised satellite image classification, Kaouter Labed 253

Embedded System for Template Matching using Swarm Intelligence, De V. Cardoso Alexandre [et al.] 256

A parallel BSO metaheuristic for molecular docking problem, Saadi Hocine [et al.] 266

One-Class Subject Authentication using Feature Extraction by Grammatical Evolution on Accelerometer Data, Mauceri Stefano [et al.] 273

A heuristic approach for standalone clinical laboratory layout design, Faramarzi Oghani Sohrab [et al.] 283

Optimizing injection blow molding by neuroevolution, Silva Hugo [et al.] 291

Reducing environmental impacts in heat exchanger networks using Life Cycle Assessment and metaheuristic optimization techniques, Pavão Leandro [et al.] 301

Energy efficient scheduling of a multi-states and multi-speeds single machine system, Aghelinejad Mohsen [et al.] 309

Meta-heuristics for global reliability optimization of solder joints in electronic devices, Hamdani Hamid [et al.] 312

Dynamic Programming heuristic for k-means Clustering among a 2-dimensional Pareto Frontier, Dupin Nicolas [et al.] 314

The Evaluation-times Constrained Optimization (ECO) Problem and Its General Solver Model, Tamura Kenichi 322

A pickup and delivery problem with multi-trips, multi- ux, multi-vehicles and break placement, Noumbissi Tchoupo Moïse Aimé [et al.] 325

Iterated Local Search for the Integrated Single Item Lot Sizing Problem for a Flow Shop Configuration With Energy Constraints, Rodoplu Melek 328

Estimation-based algorithm for a stochastic one-commodity pick-up & delivery travelling salesman problem, Hadjadj Mohamed Seddik [et al.] 331

APM-MOEA : An asynchronous parallel model for multi-objective evolutionary algorithms, Mazière Florian [et al.] 339

Quaternion simulated annealing for large-scale unconstraint continuous optimization problems, El Afia Abdellatif [et al.] 351

A new cut-based genetic algorithm for graph partitioning applied to cell formation, Boulif Menouar 361

Optimization of an Underground Water Pipeline Building using Swarm Intelligence Algorithm PSO, Bellala Djamel [et al.] 370

A Genetic Algorithm for selecting feature extraction strategy and data mining algorithm to optimize GPCR classification, Bekhouche Safia [et al.] 374

A new domain decomposition method for a reaction advection diffusion equation, Mohamed Ridouan Amattouch 385

Planet Wars: an Approach Using Ant Colony Optimization, Baldominos Alejandro [et al.] 392

A Steganographic embedding scheme using Improved-PSO approach, Yamina Mohamed Ben Ali 399

Backpropagation and PSO-GA based Optimizations for a Neural Network Classification of Engine Fault Signals, Mjahed Soukaina [et al.] 407

Customer Order Scheduling by Scattered Wolf Packs, Riahi Vahid [et al.] 409

For solving the multi-depot fleet size and mix open vehicle routing problem, Ismail Sabrine [et al.] 419

Evolutionary operators in memetic algorithm for matrix tri-factorization problem, Hribar Rok [et al.] 426

IP Assignment Optimization for an Efficient NoC-based System using Multi-objective Differential Evolution, Bougherara Maamar [et al.] 435

A modified fixed point method for biochemical transport, Mohamed Ridouan Amattouch 445

Multi-objective optimization of the integrated problem of location assignment and straddle carrier scheduling in maritime container terminal at import, Dkhil Hamdi [et al.] 450

A hybrid algorithm based on Particle Swarm Optimization and Simulated Annealing for electrical power transmission, Zemzami Maria [et al.] 474

Modelling the shortest Hamiltonian circuit problem in superimposed graphs with Distributed Constraint Optimization Problems, Bouazzi Khaoula [et al.]477

Optimization of Vehicle Routing Problem in the Context of Reverse Logistics of Handling Containers in Closed Loop, Bouanane Khaoula 481

Pricing & Lot Sizing problem in a hybrid manufacturing/ remanufacturing system with one-way substitution option, Zouadi Tarik [et al.] 483

Dynamic Simulated Annealing with Adaptive Neighborhood using Hidden Markov Model, Lalaoui Mohamed [et al.] 486

Fast Generation of Combinatorial Objects, Parque Victor [et al.] 496

Genetic Algorithms for Optimizing Nanostores' Routing in Emerging Markets, Sabiri Asmaa [et al.] 505

A cooperative multi-swarm particle swarm optimizer based hidden Markov model, Aoun Oussama [et al.] 509

Exact and multi-objective evolutionary-based approaches for process plan generation in a reconfigurable manufacturing environment, Touzout Fayçal A. [et al.] 521

A new heuristic method for a dynamic pricing and production problem, Couzon Paulin [et al.] 531

Haralick Texture Features Selection For Ultrasound Image Segmentation By Multigene Genetic Programming, Fatma Zohra Benabdallah [et al.] 534

An Energy-Efficient Permutation Flowshop Scheduling, örnek Mustafa Arslan 537

A Binary Genetic Algorithm for Solving Bi-Objective Multidimensional Knapsack Problem, Kabadurmuş özgür 548

An Agent-Based Label Propagation Algorithm for Community Detection, Fiscarelli Antonio Maria 558

Forecasting patients flow at an emergency department, Sadeghi Rezvan [et al.] 561

Single-objective Real-parameter Optimization:Enhanced LSHADE-SPACMA Algorithm, Hadi Anas [et al.] 564

Wifi emitters deployment for fingerprinting localization using genetic algorithm, Mourad-Chehade Farah [et al.] 574

Author Index **577**

first_page_proceedings.pdf **577**

Unsupervised aerial image classification using fly algorithm and the Parisian approach

Mohammed MAIZA
Laboratory of modeling and optimization of industrial systems (LAMOSI)
Faculty of Exact and Computer Sciences,
UHBC, ALGERIA
m.maiza@univ-chlef.dz

Abdelmalik TALEB-AHMED
Laboratory of Institute of Electronics, Microelectronics and Nanotechnology (IEMN)
The University of Valenciennes and Hainaut-Cambresis, FRANCE
abdelmalik.taleb-ahmed@univ-valenciennes.fr

Chahira CHERIF
Research Laboratory in Industrial Computing and Networks (LRIIR)
Faculty of Exact and applied Sciences,
Ahmed Benbella Oran 1 University, ALGERIA
cherifchahera@gmail.com

Mohamed BENYETTOU
Laboratory of modeling and optimization of industrial systems (LAMOSI)
Faculty of science, Department of computer science,
USTO, ALGERIA
med_benyettou@yahoo.fr

***Abstract*—The unsupervised classification is to look for homogeneous groups in a data set. This problem is very complex and the use of approximation algorithms is inevitable. In this paper we are interested in fly algorithm to unsupervised image classification. This new algorithm has good coding of immune systems and a variation operator of evolutionary algorithms has been combined with the Parisian approach. The latter ensures that the entire population is the solution, whose main interest is the representation of individuals of small size and rapid convergence.**

Keywords—unsupervised classification, evolutionary algorithm, fly algorithm, Parisian approach.

I. Introduction

News in the field of clustering images revolves around evolutionary approaches. Several algorithms have been applied such as genetic algorithms [6] and artificial immune systems. [7] These last two were proven performance but each of them has a defect where does the idea of combining the power of genetic algorithms and coding classes of artificial immune systems. This has already been successfully applied in several fields such as robotics and stereovision. [5] In this paper we will see that it is even possible to use it to solve the problem of unsupervised image classification. [1] [2]

II. Fly algorithm and Parisian approach

The fly algorithm [4] is a special evolutionary algorithm. It has the same variation operators that evolutionary algorithms (crossover and mutation), and adds the migration. The main interests of the fly algorithm with the Parisian approach are the reduced representation of individuals and rapid convergence, from where their utility and their popularity in real time require some robotic applications processing.

We will see that it is possible to use them to solve the problem of unsupervised classification.

A. Fly algorithm

The fly hunts for food and a mate within one to two month lifespan. When a fly decides to go for hunting, it will fly randomly (with Lévy flight motion) to find the location guided by a particular odor. While searching, the fly also sends and receives information from its neighbors and makes comparison about the best current location and fitness. If a fly has found its spot, it will then identify the fitness by taste. If the location no longer exists or the taste is ‘bitter’, the fly will go off searching again. The fly will stay around at the most profitable area, sending, receiving and comparing information at the same time. The total number of flies depends upon the number of sources.
However, since most of the flies are near to the food source location, then the next generation of flies is considered to be already close by to the potential food location.

B. Parisian approach

In most evolutionary approaches, each individual in the population represents a potential solution to the problem addressed.

This approach has proven effective for genetic algorithms, but presents a problem if the individual encoded solution is large. Manipulate a population of large number of individuals who are important to themselves can be very difficult in time and memory capacity. This problem has hampered the use of evolutionary approaches in environments where resource constraints and time are great, especially in embedded systems and robotics.

The Parisian approach explores a new possibility of coding the solution. The individual is no longer a solution, but a fragment of it and it is necessary to combine all individuals to build a solution.

There are two necessary conditions for the implementation of this approach:

- The solution can be decomposed into separate elements.
- Individuals (fragments of solution) can be judged separately by a fitness function (or objective function), and it must be representative of the "quality" of the overall solution. [5]

Later we will see how the problem of unsupervised classification can satisfy these conditions.

III. Application of Fly algorithm and Parisian approach to classification

To adapt fly algorithm with Parisian approach to the problem of unsupervised image classification, we must first meet the two conditions mentioned above: encode individuals so that the entire population is a potential solution of the problem and find a function that evaluates these individuals so that the quality of the classification is good.

A. Initialization of the population

For coding, we exploited the technique of the immune systems. We randomly create a population of '*u*' antibodies, which will be the individuals of the population P_t. Each individual (encoded with positive real numbers) represents a class. The population size is determined by the maximum number of classes.

B. Crossing

It is important to note that the method chosen of crossing the cross depends on the representation of individuals (in our work the representation is real). Thus, to calculate the new individuals (children) P_β just choose a cutoff point (that is to say a position in the chain) from which the data is exchanged between the two parents.

C. Mutation

Mutation is the second operator of variation used just after the crossing. It consists to change the individuals resulting from the crossing.

In the case of a real representation, the simplest way is to add to each individual component a realization of a standard normal distribution.

D. Migration

Migration is to introduce a set number of new individuals P_γ created randomly from the current population in order to ensure the diversity of the population.

E. Evaluation and selection of individuals for replacement

The last step of the evolutionary algorithm is the evaluation of individuals and their selections for replacement [3]. With regard to the evaluation, the used function is the following one:

$$Sim(a_j) = 1 - \frac{\sum_{i=1}^{n} d(a_j, o_i)}{n} \qquad (1)$$

Where 'o_i' the objects to be classified,
'a_j' antibodies,
'd' is any normalized distance.

After obtaining a population *P'* (parent P_t, children P_β from crossing of `*u*` parents, P_α mutations of children and P_γ from migration we need a strategy to select the `*u*` individuals which will be part of the population of the next generation (that is to say the next iteration).

Several replacements exist, one that was considered effective for our application is the elitist replacement and has a policy to keep parents which are more efficient than children without distinguishing between parents and children and keep μ best individuals from *P'*.

- Fly algorithm

```
1. Initialize a population $P_t$ of $\mu$ individuals.
2. While (stopping condition = false) do
   - Create $\beta$ individuals by crossing from $P_t$ to form the
       population $P_\beta$.
   - Create a population $P_\alpha$ with these muted individuals.
   - Generate randomly $\gamma$ individuals to form the $P\gamma$ population.
   - $P' \leftarrow P_t + P_\alpha + P_\beta + P_\gamma$
   - Evaluate all individuals from $P'$ with a fitness function and
       choose $\mu$ best ($P_t \leftarrow$ best individuals from $P'$)
End while.
```

Once the stop condition is satisfied (number of generations reached), we will have in output the '*u*' best individuals representative of class. At this stage, we calculate the distance between each pixel of the image and the '*u*' best individuals, and the assignment will be the nearest individual.

IV. Implementation

The application of fly algorithm, like any evolutionary algorithm is influenced by factors such as the form of encoding (binary or real number), the population size, the fitness function, genetic operators (crossover, mutation) .... In what follows we will discuss the parameters that are fixed by explaining the reasons, as well as those that will vary.

A. Discussion of parameters

To fix the maximum number of classes, we estimate (visually) the number of classes constituting the image, and then multiply it by two.

The used crossover model is the generalization of the method of crossing of binary chain with a single cutoff point for its ease of implementation.

With regard to the standard deviation of the normal distribution (mutation), it is important to note that despite the existing differences between individuals, their performance has to be roughly equal. This phenomenon induces premature convergence. To mitigate this risk it is better to reduce the variance using the reduced centered normal distribution (expectation = 0, standard deviation = 1).

Another parameter to fix is the number of individuals migrated. This parameter impulse on the convergence of the algorithm when to migrate a high number of individuals, in the contrary case, the diversity of the population is weakened. For this reason, it is best to migrate *'u'* individuals.

The number of generations 'GN' is the only parameter that is varied to know what happens.

B. Discussion of results

We will apply fly algorithm with the Parisian approach on an aerial image of the INCT (National Institute of Cartography and Remote Sensing) Houssine Day, Algiers, and was taken in 2010, it is a grayscale image captured by King-Air C90 aircraft with a shooting system using analog type cameras RMK TOP 30**,** representing the region of the province of FEZZARA TARF (Algeria).

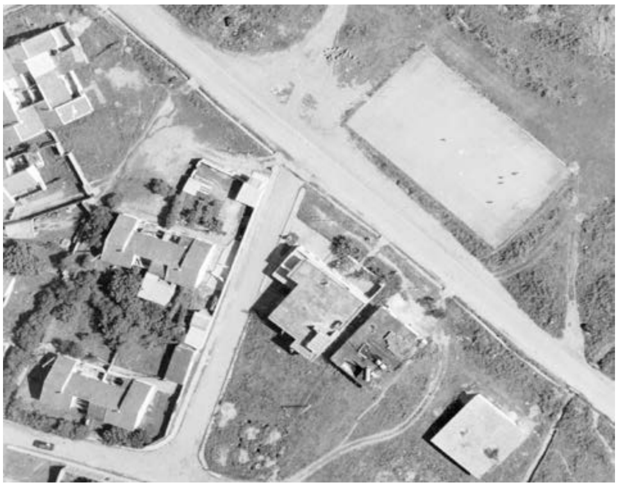


Fig. 1. Aerial image representing region of FEZZARA

TABLE I. List of Class and number of labeled samples in each class

Class Name	**Number of labled samples**
Vegetation	784
Building	291
Road	369
Shadow	103
Total number of samples	1547

1) Test 1

As a first test we will apply our algorithm while fixing the number of generation at 10. The results are given in Figure 2.

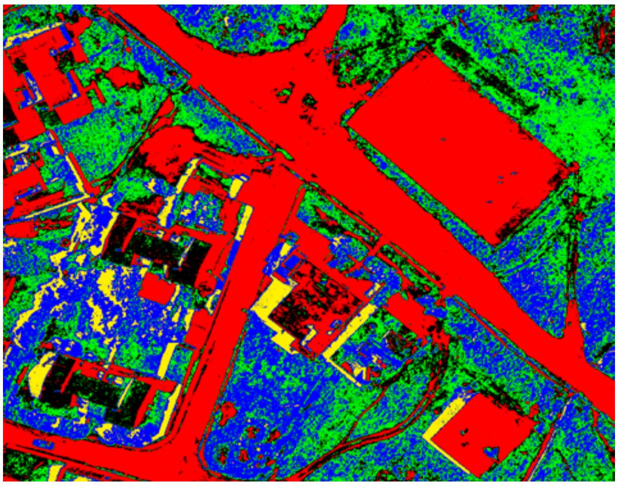


Fig. 2. Classification results with GN = 10

TABLE II. Result of classification (GN =10)

	Vegetation	Building	Road	Shadow	Total
Vegetation	**461**	95	32	8	596
Building	302	**112**	106	3	523
Road	21	43	**219**	4	287
Shadow	0	41	12	**88**	141
Total	784	291	369	103	**880**

The Table 1 shows that the number of classes that make up the image is properly recognized for the three images despite the existence of small confusion that does not exceed two percent, and some misclassified pixels that can be seen in the classified images.

This confusion may be caused by the random initialization of the population. For this reason, in the following we will keep the same initial population while varying the number of trials generations.

2) Test 2

In this essay we will fix the number of generations to 50 and then to 100.

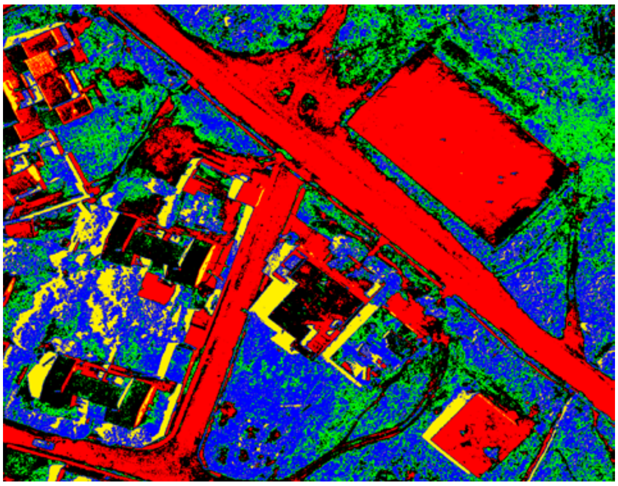


Fig. 3. Classification results with GN = 50

TABLE III. RESULT OF CLASSIFICATION (GN =50)

	Vegetation	Building	Road	Shadow	Total
Vegetation	**602**	95	32	8	737
Building	164	**192**	106	3	465
Road	18	43	**287**	4	352
Shadow	0	41	12	**94**	147
Total	784	291	369	103	**1175**

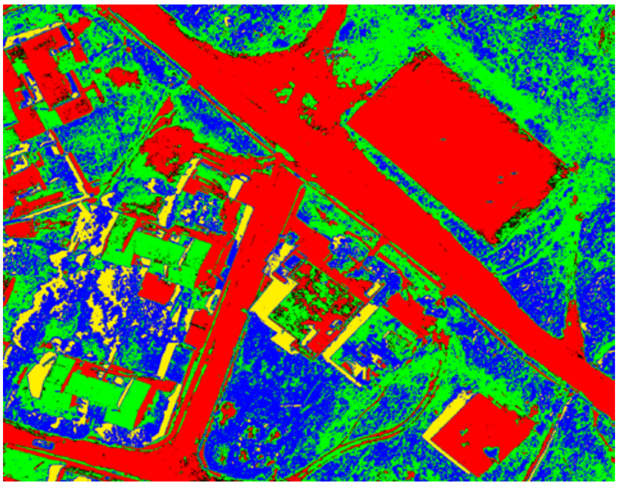


Fig. 3. Classification results with GN =100

TABLE IV. RESULT OF CLASSIFICATION (GN =100)

	Vegetation	Building	Road	Shadow	Total
Vegetation	**394**	95	32	4	525
Building	226	**89**	106	12	433
Road	161	43	**123**	19	349
Shadow	3	21	22	**52**	98
Total	784	291	369	103	**658**

These results show that the variation of the number of generations impulses on the results of two different ways: in addition increases the number of generations a better classification (GN =50) were obtained, but arriving at a number generations (GN = 100) the performance of our algorithm starts to degrade. This degradation is due to the migration operator of integrating individuals uniformly in every generation.

V. CONCLUSION

The fly algorithm with Parisian approach is a recent evolutionary approach which has the advantage of rapid convergence and a population of reduced memory size. The results have proved generally better than those of evolutionary algorithms using conventional coding. However, there are some insignificant and small confusion classes.

This problem can be resolved by varying the number of generations, but this variation can lead to a divergence caused by the migration operator by introducing individuals uniformly to every generation that keeps them from themselves into classes, so it would certainly more effective to ensure that the operator introduces individuals around groups and not uniformly.

ACKNOWLEDGENT

The authors would like to thank ANR CYCLOPE in funding this research.

REFERENCES

[1] Admane, L. et Benatchba, K. et Koudil, M. et Siad, L. et Maziz, S.''AntPart : an algorith for the unsupervised classification problem using ants.'' LMCS Laboratory. INI. 2006.

[2] Belaïd, A. ''Implémentation d'une méthode de classification non supervisée de données à l'aide des AG évolutionnaires ''. Mémoire d'ingénieur. INI. 2008.

[3] Bhandari, D. et Murthy, C. A. et Pal, S. K." Genetic algorithm with elitist model and its convergence ". *International Journal of Pattern Recognition and Artificial Intelligence*, 10(6). Pages : 731-747. 1996.

[4] Lamali,M.''Approches évolutives pour la classification non supervisée de données'' Ecole national Supérieure d'Informatique (ESI), Oued-Smar, Alger, 2009.

[5] Louchet, J. et Guyon, M. et Lesot, M-J. et Boumaza ,A. ''L'algorithme des mouches dynamiques : guider un robot par évolution artificielle en temps réel ''. *Extraction des Connaissances et Apprentissage : Apprentissage et Evolution*, vol.1– N.3. Pages 115-130. 2001.

[6] Yang, F.Y. et Lohmann, P. et Heipke, C. ''Genetic Algorithms For Multi-Spectral Image Classification'' Institute of Photogrammetry and GeoInformation, Leibniz University of Hannover, *springer*2006.

[7] Zhang, L. et Zhong,Y. et Li, P. ''An Unsupervised Artificial Immune Classifier forMulti/Hyperspectral Remote Sensing Imagery'' *IEEE Transactions On Geoscience And Remote Sensing*, VOL. 44, NO. 2, FEBRUARY 2006.

Maturation of Individuals in Evolutionary Learning

T. Calenda[1], A. Vitale[2], A. Di Stefano[2], V. Cutello[1], E-G. Talbi[3] and M. Pavone[1]

[1] Department of Mathematics and Computer Science
University of Catania
V.le A. Doria 6, I-95125 Catania, Italy
cutello@dmi.unict.it; mpavone@dmi.unict.it

[2] Department of Electric, Electronics and Computer Science
University of Catania
v.le A. Doria 6, I-95125, Catania

[3] Laboratoire LIFL, Université Lille 1
UMR CNRS 8022, Cite scientifique
Bat. M3, 59655 Villeneuve dAscq cedex, France
el-ghazali.talbi@univ-lille1.fr

Abstract. Although it is well-known that a proper balancing between exploration and exploitation plays a central role on the performances of any evolutionary algorithm, what instead becomes crucial for both is the life time with which any offspring maturate and learn. Setting an appropriate lifespan helps the algorithm in a more efficient search as well as in fruitful exploitation of the learning discovered. Thus, in this research work we present an experimental study conducted on eleven different age assignment types, and performed on a classical genetic algorithm, with the aim to (i) understand which one provides the best performances in term of overall efficiency, and robustness; (ii) produce an efficiency ranking; and, (iii) as the most important goal, verify and prove if the tops, or most, or the whole ranking previously produced on an immune algorithm coincide with that produced for genetic algorithm. From the analysis of the achievements obtained it is possible to assert how the two efficiency rankings are roughly the same, primarily for the top 4 ranks. This also implies that the worst option obtained for the immue algorithm continues to be a bad choice even for the genetic algorithm. The most important outcomes that emerge from this research work are respectively (1) the age assignment to be avoided, from which are obtained bad performances; and (2) a reliable age to be assigned to any offspring for having, with high probability, robust and efficient performances.

1 Introduction

As it is well known in the natural computing field, one of the major successful factors in evolutionary algorithms is the design and development of the exploration and exploitation mechanisms. A good balancing between these two phases is crucial since it strictly affects the efficiency and robustness of evolutionary algorithms performances. While the aim of the exploration mechanism is to search for new solutions in new regions by using the mutation operator, the second mechanism has the purpose to exploit in the best possible way all information gathered using the selection process. Both phases, hence, help the algorithm in discovering, gaining and learning new information, and, subsequently, in exploiting all gained promising regions so to generate better populations.

However, what allows to take advantage of the acquired information is truly given by how long each individual lives and in doing so influencing the evolution and maturation of the population. Besides, this lifetime affects, also, the exploration phase, allowing having a better and deep search process. Thus, the time an individual remains in the population becomes crucial in the performances of any evolutionary algorithm, and it is strictly related to the good balancing between the exploration and exploitation processes. Indeed, letting individuals live for a long time produces a dispersive search, and, then, an unfruitful learning, with the final outcome of increasing the probability to easily get trapped in local optima due to the low diversity that is generated. On the other hand, allowing a short lifetime often does not help to have enough overall learning of the knowledge discovered, and it neither allows a careful search within the solutions space, producing instead high diversity into the population, which, in turn, negatively affects the convergence towards a global optimum.

A first research work on this aspect was conducted in [3], where the authors presented an experimental study whose main aim was to understand the right lifetime of any individual/solution in order to perform a proper exploration within the search space, as well as a fair exploitation of the gained information. Such experimental analysis was conducted on an immune algorithm, whose core components are the cloning, hypermutation and aging operators.

In the cited research work, eleven different options about the lifetime of each individual were studied (see table 1), with the main goal to answer the three main questions: (i) *"is the lifespan related to the number of offspring generated?"*; (ii) *"is the lifespan related to the population size?"*; and in case of negative answer to the two previous ones, (iii) *"how long must the lifespan of an offspring be to carry out a proper exploration?"*. Once these questions were answered, an efficiency ranking was produced, from which clearly emerged that a too short lifetime is always the worst choice; whilst the best one is to let it evolve for all iterations allowed starting from scratch, i.e. assigning age 0 to each offspring. Thus, following the above described study, in this research work we want to check if the achievements produced on the Immune Algorithm (IA) are still valid, and primarily work, on a genetic algorithm (GA). Of course, what we do not expect to get the same efficiency ranking, but rather we would like to check if the top 4 for IA still appear as the top 4 for GA, even if in different ranking order, and, moreover, if the worst for IA continues to still be the worst for GA. In a nutshell, what we would like to assert with this research work is the existence (if any) of a lifespan, to be assigned to the offspring, that roughly provides robust and efficient performances, especially when high uncertainty on the problem to be solved exists; as well as which is to be avoided for sure. It is important to emphasize that such outcomes are correct and valid on all those problems that show similar landscape topologies, and similar complexities to the problem tackled.

2 The One-Max Problem

To validate and generalize the obtained results, it is crucial to develop an algorithm, which is not tailored to a specific problem, by keeping it unaware of any knowledge about the domain. As it is well-known in literature, to tackle and solve generic and complex combinatorial optimization problems, any evolutionary algorithm must incorporate local search methodologies, used as refinement and improvement of the fitness function, and this means that they have to add knowledge about the features of the problem and application domain. This, consequently, makes the algorithm unsuitable and inapplicable to any other problem, restricting then the validity of the outcomes only on such kind of problem. To overcome this limitation and make the outcomes as general as possible, in this study we tackle the classic *One–Max* (or *One–Counting*) problem [9, 2], as done in [3]. *One–Max* is a well-known toy problem, used to understand the dynamics and searching ability of a generic stochastic algorithm [8]. Although it is not of immediate scientific interest, it represents a really useful tool in order to well understand the main features of the algorithm, for example: what is the best tuning of the parameters for a given algorithm; which search operator is more effective in the corresponding search space; how is the convergence speed, or the convergence reliability of a given algorithm; or what variant of the algorithm works better [1]. It is worth emphasizing that a toy problem gives us a *failure bound*, because a failure occurs in toy problems at least as often as it does in more difficult problems. Formally, given a bit string $s = \{s_1, \cdots, s_\ell\}$ of length ℓ, the *One-Max* problem is simply defined as the task to maximize the number of 1 inside s, i.e.:

$$\text{maximize } \; f(s) = \sum_{i=1}^{\ell} s_i, \text{ with } \; s_i \in \{0, 1\}. \tag{1}$$

The choice of this simple problem, but enough complex to validate the outcomes, is therefore due mainly to the faithfully reproduction of the experimental study conducted in [3], but also because of its *"blind"* features that guarantee us to can generalize all outcomes produced.

3 The Age Assignments Studied

This research work, as well as the previous one proposed in [3], arise from observing how an algorithm (specifically an IA) can obtain considerably different performances changing only the age to assign to each offspring [7], which clearly proves how the age assignment plays a crucial and central role on the performances of the algorithm in term of success and convergence. In order to reach the goals of this research work, the same eleven age assignment types proposed in [3] have been investigated, and they are reported in table 1. In the table are reported, respectively, types and symbols used for showing and describe the results, and, in the last column, a short description of them.

Table 1. Age assignments studied.

Type	Symbol	Description
0	$[0:0]$	age zero
1	$[0:\tau_B]$	randomly chosen in the range $[0:\tau_B]$
2	$[0:(2/3\ \tau_B)]$	randomly in the range $[0:(2/3\ \tau_B)]$
3	$[0:inherited]$	randomly in the range $[0:inherited]$
4	$[0:(2/3\ inherited)]$	randomly in the range $[0:(2/3\ inherited)]$
5	$inherited$ or $[0:0]$	inherited; but if constructive mutations occur then type 0
6	$inherited$ or $[0:\tau_B]$	inherited; but if constructive mutations occur then type 1
7	$inherited$ or $[0:(2/3\ \tau_B)]$	inherited; but if constructive mutations occur then type 2
8	$inherited$ or $[0:inherited]$	inherited; but if constructive mutations occur then type 3
9	$inherited$ or $[0:(2/3\ inherited)]$	inherited; but if constructive mutations occur then type 4
10	$inherited - 1$	same age of parents less one

It is possible subdivide the age assignment types in three different efficacy groups: (1) the fixed ones (`type0` and `type10`); (2) the random ones (from `type1` to `type4`); and (3) the constructive change ones (from `type5` to `type9`). In the first group to each offspring is assigned the same age for everyone; in the second one, instead, to each offspring is assigned a random age, ensuring however evolve at least for a fixed number of generations, except for the `type1` that in the worst case will assign age τ_B. Besides, `type3` and `type4` differ from the previous two as each offspring will have the same age (in worst case) or less than its parent (labelled as *"inherited"*). Finally, all age types included into the last group produce higher diversity in the population than the others, and encourage those offspring that appears to be promising. Indeed, to each clone is assigned the same age of the parent at first, which generates high turnover degree in the population, but if after the genetic (crossover and/or mutation) operators, the fitness of the offspring is improved, then its age is updated in order to have more evolutionary time.

4 The Genetic Algorithm

In this research work we have developed a classical Genetic Algorithm (GA), one of the most well known evolutionary algorithms, which take inspiration from genetics and natural selection. The genetic algorithms represent an efficient and robust algorithmic class in search and optimization, thanks to their ability in fruitfully exploration of search space, and efficiently exploiting of the promising regions. This class of algorithms is based on three main evolutionary operators, such as the (i) *recombination*, where each generated offspring inherits some characteristics from the two parents; (ii) *mutation*, which introduces diversification in the offspring with respect to the parents; and (iii) *selection mechanism*, through which the individuals for mating pool are selected. The first two operators represent the exploration phase of new search points, whilst the last one helps the algorithm to exploit information learned in order to generate improved progenies (exploitation phase). Note, however, that in the GAs the mutation operator plays a secondary role than the other two.

For achieving the set out goals in this research work, we have appropriately adapted the developed genetic algorithm so that a given age is assigned to each generated individual at each time step. In this way, it is determined the lifespan of each offspring inside the population, whose evolution starts from such assigned age (see section 3), which is increased by one at each generation, until reaching the maximum age allowed (τ_B, an user-defined parameter). Further, the aging operator as designed in [3, 7], was developed and included into this proposed GA [5, 6].

The algorithm starts with the creation of the initial population ($P^{(t=0)}$) of size pop_size, by generating random solutions, i.e. bit strings of length ℓ, using uniform distribution, and thus the next step is the evaluation of the fitness of each individual, using the function *Compute_Fitness*($P^{(t)}$). Whereupon, begins the evolution of the algorithm whose considered termination criterion is the reaching of the maximum allowed number of fitness function evaluations (T_{max}). The algorithm terminates in advance the execution if the global optimal solution is found. Note that age zero is assigned to each newly created individual, regardless of the age assignment chosen (see the age types in section 3). A summary of the developed Genetic Algorithm is presented in the pseudocode shown in Algorithm 1.

Algorithm 1 Pseudo code of GA

```
Genetic Algorithm ($pop\_size, p_c, p_m, \tau_B, Elitism$)
$t \leftarrow 0$
$FFE \leftarrow 0$
$P^{(t)} \leftarrow$ Create_Initial_Population($pop\_size$);
Compute_Fitness($P^{(t)}$)
$FFE \leftarrow FFE + pop\_size$
while ($FFE < T_{max}$) do
    Increase_Age($P^{(t)}$);
    $P^{(parents)} \leftarrow$ Roulette_Wheel_Selection($P^{(t)}, pop\_size$)
    $P_c^{(kids)} \leftarrow$ Crossover($P^{(parents)}, p_c$)
    $P_m^{(kids)} \leftarrow$ Mutation($P_c^{(kids)}, p_m$)
    Compute_Fitness($P^{(t)}$)
    $FFE \leftarrow FFE + pop\_size$
    $({}^aP^{(t)}, {}^aP_m^{(kids)}) \leftarrow$ Aging($P^{(t)}, P_m^{(kids)}, \tau_B, Elitism$);
    $P^{(t+1)} \leftarrow (\mu + \lambda)$–Selection(${}^aP^{(t)}, {}^aP_m^{(kids)}$);
    $t \leftarrow t + 1$
end while
```

Inside to the iterative loop the first step is to increase the age of each individual by one, becoming older than a generation (function *Increase_Age(P^t)*). Hence, the selection of the individuals for the mating is applied using the classical *Roulette-Wheel-Selection* model [4, 10], which basically select the individuals with a probability proportionally to their fitness: the higher the fitness, the higher the likely is that they will be selected. At this step a new population $P^{(parents)}$ of size pop_size is created that contains all selected individuals, from whose mating will be produced the offspring. In particular, from a pair of individuals are generated a pair of offspring.

Afterwards, the parents population is then undergo to the recombination phase, with a probability p_c, which generates the population of the new offspring ($P_c^{(kids)}$). In this work we have developed the classic *Uniform Crossover*, through which each element (gene) of the offspring is randomly selected by both parents; i.e. the parents will contribute equally to generating their own descendants. Thanks to this kind of recombination operator each offspring will have 50% genes from the first parent, and the other 50% from the second one. Once $P_c^{(kids)}$ is produced, each chromosome is mutated with a p_m probability. The mutation used in our study is the well known *bit-flip mutation*, which - if applied - randomly select a gene s_i ($\forall\ i = 1, ..., \ell$) in the chromosome s, and inverts its value (from 0 to 1, or from 1 to 0). The mutated chromosomes produce a new population, labelled $P_m^{(kids)}$ (see Algorithm 1), containing the final offspring generated. When an offspring is created, to it is assigned an age that affects its lifespan inside the population. Starting from this age each chromosome will evolve until to reach a maximum age allowed (τ_B, a user-defined parameter), after which it will be removed from the population by the aging operator. Age assignment, and the aging operator have the main purpose to keep high the diversity into the population in order to avoid premature convergences and then reduce the probability to get stuck in local optimum. Therefore, choosing the age to be given plays a crucial role in the performances of the algorithm, since the evolution and maturation of the solutions depend strictly on this. Note that in general, the crossover and mutation operators do not affect the age of any new individual, except for the $5 - 9$ options of the age assignment types (see table 1, section 3), where the assigned age is updated only if its fitness value is improved. This happens and it is computed in the function *Compute_Fitness($P^{(t)}$)*.

As described above, in our GA was included the aging operator in order to achieve the determined objectives, and whose main task is to help the algorithm in jumping out from the local optima, producing high diversity into the population and avoiding, consequently, premature convergences. It simply eliminates the old chromosomes from the two populations $P^{(t)}$ and $P_m^{(kids)}$. Every individual is allowed to mature for a fixed number of generations: as soon as it reaches age $(\tau_B + 1)$, it is removed from the population of belonging regardless of the fitness value, included the best solution found so far. The parameter τ_B indicates the maximum number of generations allowed to any chromosome to stay into the population. An exception, however, is allowed only for the best solution found so far, that is the global best solution found is always kept into the population, even if it is older than $\tau_B + 1$. Such exception is called *Elitism Aging operator*. This variant helps the algorithm keep track of the most promising region - which would otherwise be lost - and whose exploitation instead might be useful in solving some specific kinds of problems. The boolean variable $Elitism$ (Algorithm 1) controls the activation of the variant elitism aging operator.

Unlike to classic GA, where usually the offspring replace the parents for the next generation, in this work the new population $P^{(t+1)}$ is created by using the $(\mu+\lambda)$-*Selection operator*, which selects the best pop_size survivors to the aging step from the two populations ${}^aP^{(t)}$ and ${}^aP_m^{(kids)}$. This operator, simply, selects the best pop_size chromosomes from the offspring set – created by the crossover and mutation operators – and the old parent chromosomes guaranteeing monotonicity in the evolution dynamics. Nevertheless, due to the aging operator, it could happen that the number of survivor chromosomes (pop_size^1) is less than the input population size (pop_size). If so, the selection operator randomly generates $(pop\_size - pop\_size^1)$ new individuals. This step is called *Birth phase*.

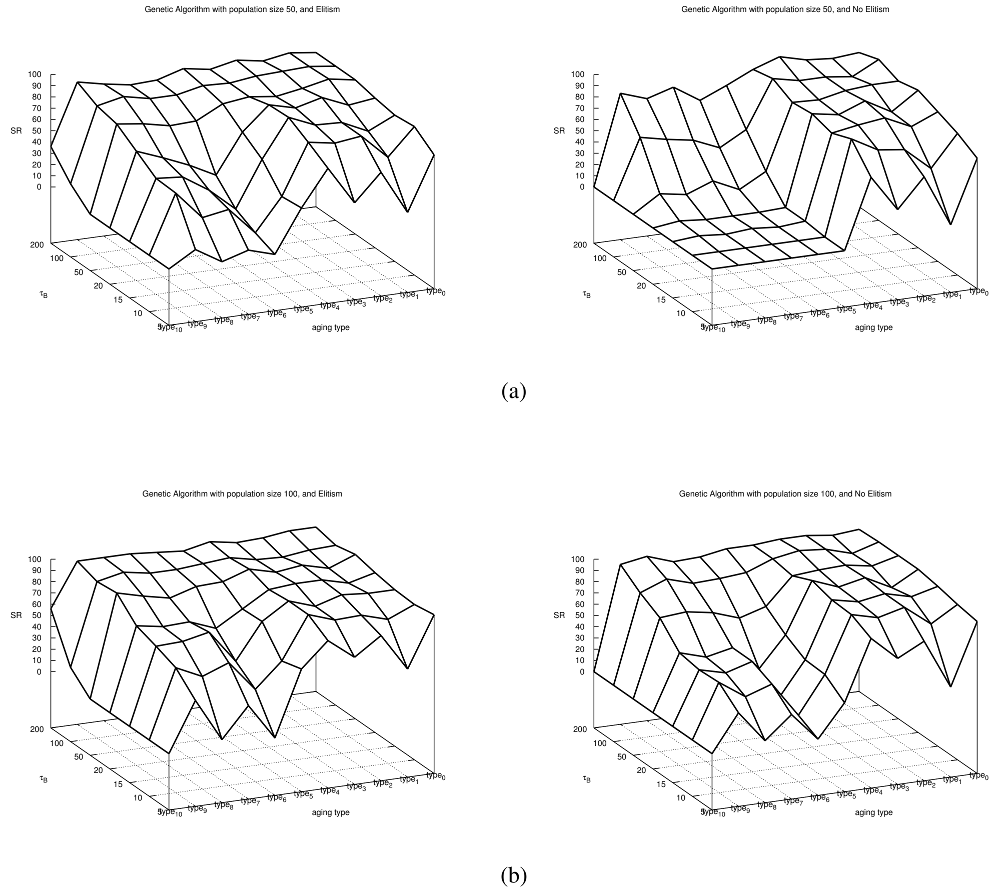
5 Experimental Results

In this section we present all outcomes obtained in our experimental study in order to (i) understand which between the age assignment types studied is more robust and efficient in the overall; (ii) produce an efficiency ranking between them; and, most important, (iii) to check if the best 4 of the efficiency ranking obtained for IA [3] still appear in the top 4 for GA, although in different rank order. Besides, it becomes also interesting to observe if the worst assignment - or the two worst - for IA continue to be so also for GA. With the outcomes of this research work, we want to give a rough indication on the needed lifetime to a solution to have a proper balancing between exploration and exploitation in order to maximize the evolutionary learning, and which one instead to not consider. Further, having developed an algorithm not tailored to a specific problem, and performed our study in an unaware way about any domain knowledge, the outcomes obtained will be even more effective when high uncertainty exists to be managed. However, the existence of one or more age assignments in common between IA and GA allow us also to provide a reliable lifespan, which, with high probability, leads to efficient and robust performances on similar features problem, and on evolutionary algorithms, in general.

In order to reach our goals, we have then used the same experimental protocol proposed in [3], except for the string length, which was instead fixed to $\ell = 2000$. This setting is due to the difficulty of the algorithm in solving the problem for higher values. Therefore, all age assignment options in table 1, have been studied varying $pop\_size = \{50, 100\}$, $\tau_B = \{5, 10, 15, 20, 50, 100, 200\}$, and setting $Tmax = 10^5$ as termination criteria (i.e. the maximum number of fitness function evaluations). Each experiment was computed on 100 independent runs. Both variants of GA have been performed: *elitism* and *no_elitism*. Regarding GA parameters, after several preliminary experiments (not included in this paper), the probability to apply the crossover (p_c) and mutation (p_m) operators have been set respectively to $p_c = 1$ and $p_m = 0.4$, for all experiments presented.

In figure 1 are showed the efficiency surfaces produced by varying τ_B and age assignment types, and evaluate with respect their success rate (SR), that is how many time the optimal solution is found in 100 runs. Both GA variants are showed for $pop\_size = 50$ (first row), and $pop\_size = 100$ (last row): the *elitism* version in the left plots, and the *no_elitism* in the right ones. Inspecting these surfaces, appear clearly in all plots how the last age assignment option (`type10`) shows the worst performances at any values of τ_B and pop_size, unable almost always in finding the optimal solution (except for the elitism variant with high τ_B values). Those who instead show better performances in the overall are `type0` and `type4`, which exhibit more robust and efficient performances, regardless of the parameters used. It is important to highlight how, for low τ_B values, the age assignment options from `type5` to `type9` (constructive change group; see table 1) produce the lower regions of SR, and this is due to their characteristic to produce high diversity into the population, which means high turnover degree and, therefore, a greater use of the *Birth* operator. This appears more prominent in the *no_elitism* variant, and especially for low *pop_size* values, where diversity becomes more pronounced. In plot, it is also possible to see how any age assignment becomes irrelevant on the performances for high τ_B value ($\tau_B = \{100, 200\}$), whose values are very close to have an infinite life.

The same results are also, and better, presented in tables 2 and 3. In each row are showed the success rate (SR) obtained, and AES (line below), i.e. the average number of fitness function evaluations to reach the optimal solution. Of course, AES is not null when $SR \neq 0$, i.e. if the optimal solution was found at



(a)

(b)

Fig. 1. Efficiency surfaces produced for *elitism* (left plots) and *no_elitism* (right plots) variants for $pop\_size = 50$ in plots (a), and $pop\_size = 100$ in plots (b).

least once. In particular, the results presented in table 2 have been obtained with $pop\_size = 50$, whilst the ones in table 3 with $pop\_size = 100$. Analysing both tables, we have clearly the confirmation that `type10` is unable to reach the optimal solution, except for the *elitism* version with setting high values of τ_B ($\tau_B = \{100, 200\}$). Further, it is interesting to note that increasing the population size (pop_size) helps GA to increase the SR (as we expect) but it doesn't affect nor alter in any way, the overall influence of the age assignments on the performances of GA, producing approximately the same efficiency rankings. This statement answers, and confirms, as also claimed for IA, that the right lifespan for which each individual must evolve is not related to the population size considered.

Inspecting table 2, considering the *elitism* variant, it is possible to see how the algorithm finds the optimal solution, at least once, with all age assignments considered, except for `type10` ($\forall\ \tau_B < 100$); `type6` and `type8` when the τ_B value is low; unlike of the *no_elitism* version where GA struggles to reach the optimal solution, and this is what we expect. However, one of our goal is to understand which age assignment type provides more robust and efficient performances. Therefore, from this point of view, it is possible to assert that `type0` and `type4` seems to be more robust than the others, guaranteeing then more reliability, whilst `type10` and `type6` continue to be the worst ones. The same statement can also be made for the *no_elitism* version. From table 3 is possible to see how the SR increases on all experiments, allowing the achievement of the optimal solution, where GA failed in the previous table. Also on these experiments `type0` and `type4` appear to be more robust and efficient than the others. Comparing these two age assignments types, it is possible to assert that, in the overall, `type0` shows more reliability than `type4`, since its use allows GA to obtain, approximately, the same, and high, success rate both in the *elitism*, and *no_elitism* variants (`type0`: 82.14 vs. 83 for $pop\_size = 50$, and 92 vs. 92 for $pop\_size = 100$ – `type4`: 83.86 vs. 79.86 for $pop\_size = 50$, and 91.57 vs. 91.71 for $pop\_size = 100$).

Table 2. GA on *One–Max* problem with $\ell = 2000$. The results have been obtained by setting: $pop\_size = 50, p_c = 1, p_m = 0.4$, with and without Elitism.

type	$\tau_B = 5$	$\tau_B = 10$	$\tau_B = 15$	$\tau_B = 20$	$\tau_B = 50$	$\tau_B = 100$	$\tau_B = 200$
				Elitism			
0	69% 90741.44	82% 86409.38	80% 87103.44	83% 86652.88)	87% 84676.01	87% 82797.44	87% 83204.83
1	21% 98724.99	58% 94472.81	69% 92170.46	80% 88594.79	91% 85552.17	90% 82780.77	90% 83890.19
2	60% 94039.49	80% 87698.82	80% 86159.5	90% 84839.10	86% 84058.78	89% 84007.95	86% 84330.07
3	36% 97341.97	77% 89434.23	81% 86796.83	84% 86317.53	82% 85648.48	88% 84421.64	90% 83416.20
4	70% 91853.51	81% 87440.62	91% 85747.82	90% 83773.17	85% 84213.71	85% 83853.50	85% 84004.85
5	38% 96661.74	36% 96847.64	57% 93269.04	69% 90052.98	82% 87376.17	86% 84609.75	89% 84330.95
6	0%	7% 99763.66	16% 98820.08	34% 97435.21	70% 89703.56	81% 86601.79	82% 85642.79
7	7% 99723.64	31% 97398.44	36% 96937.86	46% 94309.49	69% 89642.67	81% 86399.53	81% 85410.66
8	0%	27% 98454.84	51% 94153.06	56% 92745.16	74% 87680.47	86% 87118.12	85% 83791.32
9	14% 99273.65	52% 93683.66	53% 94559.11	65% 91631.61	77% 87691.02	81% 85953.13	90% 83402.34
10	0%	0%	0%	0%	0%	15% 98903.89	36% 96232.28
				No Elitism			
0	66% 90771.76	76% 87042.24	83% 86277.61	90% 85208.42	86% 85111.56	93% 82111.92	87% 83868.15
1	10% 99486.34	48% 95815.82	73% 92056.89	74% 90457.19	86% 85435.83	81% 86840.81	85% 83652.09
2	56% 94334.12	68% 88144.89	84% 87116.65	87% 84443.68	84% 86296.26	84% 85577.26	87% 84776.40
3	30% 97689.88	71% 89510.53	81% 86619.09	78% 86543.17	79% 85487.73	84% 85086.12	93% 83041.88
4	62% 92538.22	84% 86222.68	77% 87402.21	82% 86827.35	80% 85629.22	89% 84230.84	85% 83664.17
5	0%	0%	2% 99666.75	2% 99557.23	22% 95892.60	47% 90973.89	74% 84776.14
6	0%	0%	0%	1% 99921.99	9% 98130.29	34% 92421.01	64% 87918.47
7	0%	0%	2% 99601.45	1% 99673.91	20% 95497.91	44% 89819.80	79% 83756.32
8	0%	0%	0%	1% 99911.66	11% 97763.70	48% 91077.56	72% 86381.38
9	0%	0%	2% 99607.87	3% 99240.23	15% 96522.20	53% 88524.80	80% 85093.18
10	0%	0%	0%	0%	0%	0%	0%

In figure 2 is showed the histograms produced by the average of the success rates for each age assignment type. The left plot shows the average SR produced using the *elitism* variant, whilst the right plot those produced by the *no_elitism* version. Such results have been produced averaging on all τ_B values. Thanks to these plots becomes easy to produce the efficiency ranking for each variant, which seem to be the same in the overall, except for the first position: for the *elitism* variant appears `type4` on the first rank; whilst for the *no_elitism* version the top one becomes `type0`. From the third position onwards, the two efficiency ranking seems to be the same, respectively: $3)$ `type2`, $4)$ `type3`, $5)$ `type1`, $6)$ `type5`, $7)$ `type9`, $8)$ `type7`, $9)$ `type8`, $10)$ `type6`, and $11)$ `type10`.

Since it is important to produce an efficiency ranking regardless of variants used, in figure 3 we show the average success rate ($\widehat{SR}$) produced by both the variants on all experiments performed. From this plot, emerge `type0` as the best ($\widehat{SR} = 87.27\%$), followed by `type4` ($\widehat{SR} = 86.75\%$), `type2` ($\widehat{SR} = 84.89\%$), `type3` ($\widehat{SR} = 80.61$), and `type1` ($\widehat{SR} = 73.93\%$). From these overall results appear clear as the last type, the `type10`, is always the worst with an overall average success rate considerably low with respect the others ($\widehat{SR} = 4.39\%$), and never reaching the optimal solutions in *no_elitist* variant (Fig. 2, plot b). Just to note that the penultimate in the efficiency ranking between the age assignment types (*type6*) produces an overall average success rate of $\widehat{SR} = 38.75\%$.

Once generated the efficiency ranking for GA, it is possible to compare the outcomes obtained with the ones produced by IA in [3], in order to reach the third and most important goal of this work. Analysing the comparisons, then, it is possible to clearly assert that the top 4 types produced by IA are still in the top 4 of

Table 3. GA on *One–Max* problem with $\ell = 2000$. The results have been obtained by setting: $pop\_size = 100, p_c = 1, p_m = 0.4$, with and without Elitism.

type	$\tau_B = 5$	$\tau_B = 10$	$\tau_B = 15$	$\tau_B = 20$	$\tau_B = 50$	$\tau_B = 100$	$\tau_B = 200$
Elitism							
0	91% 82312.79	88% 79513.60	89% 80284.09	92% 78748.99	96% 76946.22	92% 77181.40	96% 76713.31
1	46% 96226.19	78% 89019.29	89% 84632.08	86% 84135.64	87% 81507.88	91% 78715.07	96% 78471.94
2	79% 86728.32	85% 80317.57	91% 80431.61	92% 78505.86	93% 78550.34	88% 78540.41	93% 77438.99
3	63% 91211.17	84% 84436.42	84% 82546.80	92% 80466.22	90% 78691.13	89 78144.62	92% 78627.42
4	81% 86788.92	92% 81076.87	92% 79704.66	90% 80032.76	96% 77980.19	94% 77136.83	96% 75580.17
5	59% 93334.05	54% 92368.72	77% 88048.81	76% 86315.99	88% 81170.38	92% 76130.42	91% 77768.30
6	1% 99989.08	32% 97195.04	45% 95906.89	62% 91813.69	87% 83105.46	90% 80479.18	93% 78291.70
7	33% 97217.75	59% 90119.24	74% 89718.72	71% 87589.66	80% 83268.17	90% 78213.40	95% 78284.81
8	6% 99720.49	50% 94409.41	69% 88542.49	70% 89541.02	85% 82873.96	94% 78983.65	95% 78189.25
9	44% 95440.97	61% 91484.43	68% 87646.74	74% 87221.28	91% 78151.71	89% 79294.21	95% 77613.35
10	0%	0%	0%	0%	0%	16% 98535.69	56% 93547.39
No Elitism							
0	85% 83264.41	88% 79521.29	92% 78683.06	95% 78733.66	95% 77829.72	95% 77960.86	94 79148.00
1	30% 98132.36	74% 90100.46	86% 84718.40	82% 84605.64	89% 80768.79	88% 80283.13	92% 79894.21
2	74% 88613.47	92% 80472.08	92% 79932.29	94% 78718.43	93% 77271.95	95% 78857.49	95% 76172.57
3	62% 94039.55	86% 84023.41	88% 81425.63	89% 79838.93	92% 77831.94	97% 78117.67	95% 78376.93
4	83% 84655.22	92% 80158.74	93% 79879.89	98% 77569.49	90% 78153.02	93% 77655.90	93% 77784.34
5	29% 95268.10	42% 92692.33	43% 91418.28	53% 88893.36	66% 84100.89	84% 79270.36	93% 78672.59
6	0%	9% 99385.86	12% 98633.38	26% 96286.96	59% 87265.87	82% 80200.76	89% 79989.41
7	25% 96783.49	40% 92715.84	48% 91968.87	48% 90444.85	67% 83522.93	81% 79741.93	88% 79447.61
8	5% 99780.98	32% 95006.62	41% 92756.55	43% 91618.77	71% 82310.38	88% 78390.08	96% 77734.65
9	33% 95163.12	48% 91905.64	47% 91211.35	50% 88393.57	69% 84288.04	79% 80440.52	92% 77513.32
10	0%	0%	0%	0%	0%	0%	0%

the efficiency ranking produced by GA, respecting exactly the order of the first two positions. Further, the worst age types on IA continue to be the worst even on GA, and in particular in the last two positions appear `type 6` and `type 10` respectively. Their bad performances are due to the high diversity they produce, not allowing a relevant lifetime to perform a good exploration. Finally, this research work, having found a common efficiency ranking between IA and GA, provides a reliable individuals maturation time in order to optimize the evolutionary learning and yield robust and efficient performances on those problems that show similar landscape topologies, and similar complexities to the problem considered.

6 Conclusion

In this paper we show how the age assignment, i.e. how many generations an offspring must remain into the population, plays a crucial role on the performances of any evolutionary algorithm, since it is strictly related to a correct balancing between the exploration and exploitation mechanisms. In this research work we present an experimental study focused on understanding the right maturation time of each solution in order to optimize the evolutionary learning, which means to perform a careful search process, and take advantage of the information discovered as best as possible. Since this experimental study is based on a previous one, conducted on an immune inspired algorithm, the main goal of this paper is to check and prove that the achievements previously obtained continue to be valid, and work, on a genetic algorithm. In this way, we can assert the existence of a reliable lifespan able to provide, with high probability, robust and efficient performances for any population-based algorithm, especially in uncertainty environments.

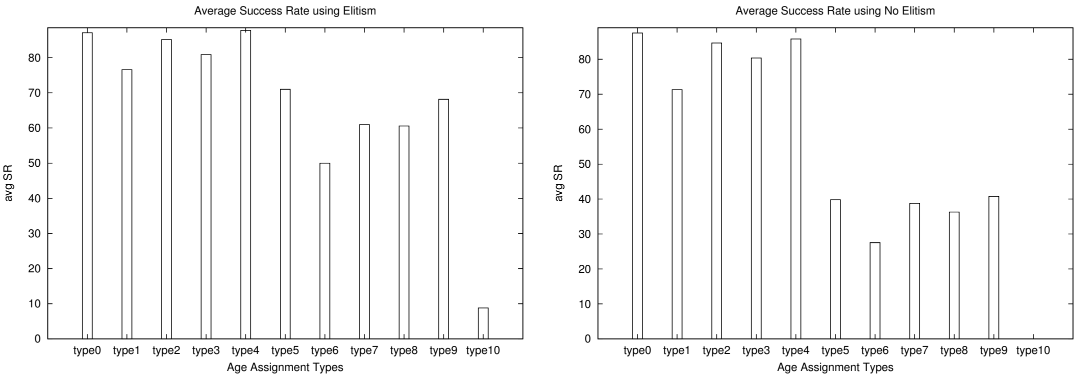


Fig. 2. Average Success Rate over all performed trials for the *elitism* (left plot) and *no_elitism* (right plot) variants.

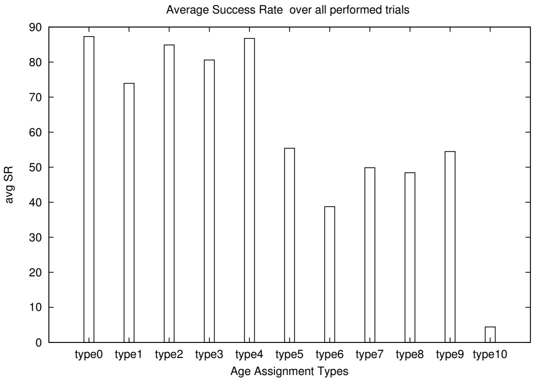


Fig. 3. Average Success Rate over all performed trials.

A classical genetic algorithm has been developed, based on the genetic operators: *roulette-wheel-selection*; *uniform crossover* and *flip mutation*. The GA was properly adapted in order to reach the set out achievement, adding an age assignment for each chromosome and introducing the aging operator, the use of which helps the algorithm to escape from local optima. Eleven different age assignment types have been considered in our study, each of which affects on the algorithm performances in different way: *static ones*; *random ones*; and *constructive change ones*. Analysing the achievements obtained in this work, and comparing them with the ones previously obtained by IA, it is possible to assert as the previous top 4 continue to be in the best 4 positions of the new efficiency ranking, keeping exactly the same order in the first two positions. In the overall, it is possible also to say that the two efficiency rankings are approximately the same, except 2-3 positions that alternate each other. This then imply that the worst, or the two worst between the age assignment types in IA continue to be the worst also in GA.

Having found a common efficiency ranking between IA and GA, points out the existence of a reliable lifetime to be assigned to each individual - for any population-based algorithm - that with high probability guarantees efficient and robust performance, especially when many information on the problem are not known a priori. Of course these statements are valid on all those problems that show similar complexities and similar landscape topologies to the problem tackled. In light of this, as future work, we want to test and perform the same experimental study on a mathematical model, that is the NK-Model, which is able to produce *"tunable rugged"* fitness landscapes. In this way we can test the achievements produced on different roughness level of the landscape.

References

1. V. Cutello, A. G. De Michele, M. Pavone: "*Escaping Local Optima via Parallelization and Migration*", VI International Workshop on Nature Inspired Cooperative Strategies for Optimization (NICSO), Studies in Computational Intelligence, vol. 512, pp. 141–152, 2013.

2. V. Cutello, G. Narzisi, G. Nicosia, M. Pavone: “*Clonal Selection Algorithms: A Comparative Case Study using Effective Mutation Potentials*”, 4th International Conference on Artificial Immune Systems (ICARIS), LNCS 3627, pp. 13–28, 2005.
3. A. Di Stefano, A. Vitale, V. Cutello, M. Pavone: “*Document How long should offspring lifespan be in order to obtain a proper exploration?*”, 2016 IEEE Symposium Series on Computational Intelligence (SSCI), INSPEC number 16670548 2016, pp. 1–8, 2016.
4. D. Goldberg: ”*Genetic Algorithms in Search, Optimization, and Machine Learning*”, Addison-Wesley publishing company, 1989.
5. N. Kubota, T. Fukuda: ”*Genetic Algorithms with Age Structure*”, Soft Computing, vol. 1, pp. 155–161, 1997.
6. A. Ghosh, S. Tsutsui, H. Tanaka: ”*Individual Aging in Genetic Algorithms*”, Australian and New Zealand Conference on Intelligent Information Systems (ANZIIS), INSPEC number 5534628, pp. 276–279, 1996.
7. M. Pavone, G. Narzisi, G. Nicosia: ”*Clonal Selection - An Immunological Algorithm for Global Optimization over Continuous Spaces*”, Journal of Global Optimization, vol. 53, no. 4, pp. 769–808, 2012.
8. A. Prugel-Bennett, A. Rogers: “*Modelling Genetic Algorithm Dynamics*”, Theoretical Aspects of Evolutionary Computing, pp. 59-85, 2001.
9. J. D. Schaffer, L. J. Eshelman: “*On crossover as an evolutionary viable strategy*”, 4th International Conference on Genetic Algorithms, pp. 61–68, 1991.
10. E. G. Talbi: “*Metaheuristics: from Design to Implementation*”, John Wiley & Sons, 2009.

Multi Objective Optimization and Bionic Optimization Strategies in Engineering Design

R. Steinbuch

RRI, Reutlingen University, Alteburgstr. 150, 72762 Reutlingen, Germany
rolf.steinbuch@reutlingen-university.de

Keywords: Multi Objective Optimization, Bionic Optimization, Pareto front, meta-models

1 Introduction

Bionic Optimization is an elementary component of the metaheuristic research. In engineering design, Evolutionary Strategies and Genetic Algorithms are the most favored approaches, but we use other ideas as well. The central weakness of the classical version of these approaches is the restriction to deal with unique goals and non-scattering parameters. One deals with both problems by specific expansions of the basic methods, the robustness and reliability analysis on the one hand, the Multi Objective Optimization (MOO) on the other hand. It would be preferable to propose a combined, unified approach to solve both extensions simultaneously following ideas proposed e.g. by Wang [1]. However, we run into the field of very large computation sets and very complex basic theory, if we follow this road. Therefor we restrict ourselves here in this presentation to the question of MOO. We introduce the basic terms and explain some solution approaches using two elementary examples.

2 The Expanded Optimization problem

We want to expand the classical optimization problem of finding the best values of a given scalar function by varying its free parameters without violation of boundary conditions or other constraints to MOO problems. We propose formulations that follows the ideas outlined by Gekeler [2] and Wang [1], but have many other parents as well. Assume a non-scalar function

$$\mathbf{r} = f(\mathbf{x}) \tag{1}$$

Rewrite it in a more explicit form

$$(r_1, r_{2,..}r_n) = f(x_1, x_{2,..}x_m), \qquad x_{i,min} \leq x_i \leq x_{i,max},\ i = 1..m \tag{2}$$

Here the r_j, $j = 1..n$ are the responses, the $x_{i,}$ $i = 1\ldots m$ stand for the free parameters, of the system. We differentiate two types of responses

$$(r_1, r_{2,..}r_n) = (s_1, s_{2,..}s_{k,}c_1, c_{2,..}c_l) \tag{3}$$

where the s_i, $i = 1\ldots k$ are the different goals to be optimized, while the constraints or restrictions c_i have to follow sets of d conditions

$$g_p(c_1, c_{2,..}c_{l,}x_1, x_{2,...}x_m) \leq 0, \qquad p = 1...d \tag{4}$$

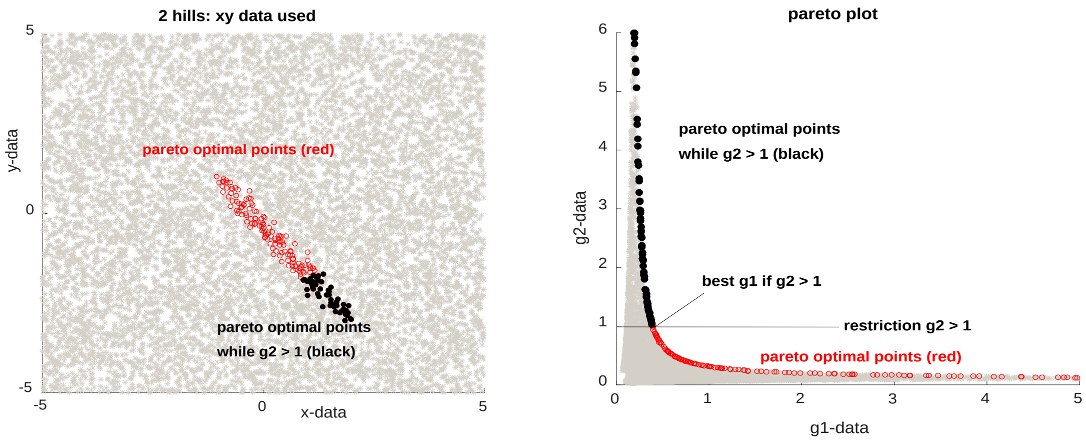
The optimization task is to find the best values of the objectives s_1, $s_{2,..}s_{k,}$ within the restrictions and the boundary conditions acceptable range. Evidently, there is no absolute and undoubted definition of the optimum of the vector assembled by the s_i. We want to discuss this problem in some more detail neglecting the fact, that all input data are always scattering as mentioned before. This would lead to an expansion into the field of reliability and robustness, which requires even more elaborated studies.

3 Solving MOO problems

To solve MOO problems it is convenient to use two main approaches:

3.1 *Reduction to one combined goal or objective* (Single Objective Optimization, SOO) is the most often used way to handle MOO problems. A very simple way to go this path is to remove all but one entry from the objectives list s_i (eq.3) and interpret them as additional restrictions c_q, whose specific values now have to be defined. This definition is done by the user's preferences, a subjective decision-making. Another way to find a SOO solution is to build a unique goal s_{unique} as a function of the given goals s_i, e.g. by a weighted sum of these goals. The definition of such a function again is up to the users preferences. We use both ideas with much success. There exist many variants of this approach.

3.2 *Pareto fronts* are sets of solutions where for all objectives s_i holds, that the increase of the value of one of them causes a decrease of the values of all other objectives. Fig. 1 depicts such a Pareto-front approach for a simple 2D-problem (Fig. 2). Fig. 1a) shows the positions of the best solutions found (red and black) in the (x,y) space of allowable parameters, Fig. 1b) plots the Pareto front as $goal_2$ vs. $goal_1$ (red and black points). When we found the Pareto front, we have a set of preferable solutions. The users might decide which element of this set they prefer. This selection of one specific solution is not part of the optimization process. It is as subjective as the switching of goals to restrictions proposed before.

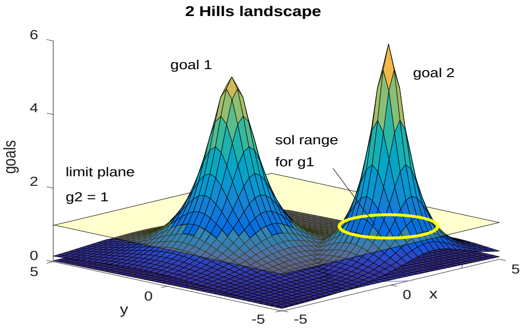


a) Random points in (x,y) space, Pareto best marked   b) Pareto front: goal1 vs goal2

Fig.1: Set of parameter combinations (a) and Pareto front (b) of the two-hill example presented below.

4 Examples of MOO problems

To explain the MOO solving strategies and demonstrate the problems occurring we propose two examples, a landscape with two hills (Fig. 2) and a beam under bending (Fig.3).



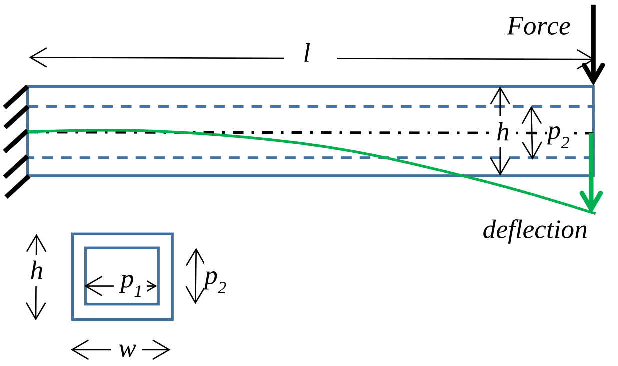
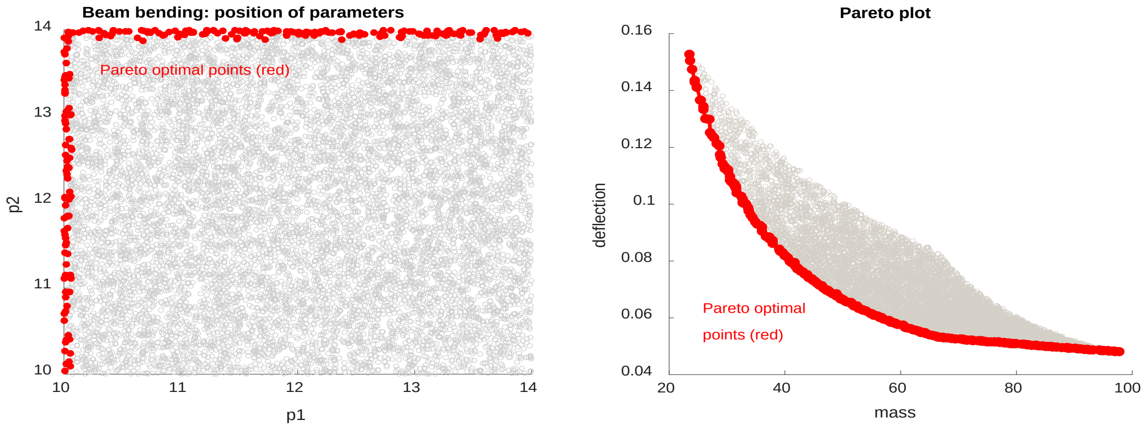


Fig. 2: Two Hills as different goals in a MOO

Fig3: Beam under load: minimize mass and deflection

For the two-hill problem the two goals are given by the heights g_1 and g_2 of the hills at a given position inside the defined (x,y)-region. Fig. 1 already showed the position of the best solutions and the Pareto front. We found the values of these two plots by a random search, indicated by the many points (grey) on both plots. We could have found a good guess of the Pareto front by introducing some required minimum values of g_2 (or g_1) and then find the corresponding optimum off g_1 (or g_2) indicated in Fig. 1, where we introduced the restriction $g_2 > 1$ and maximize g_1 or vice versa.

Fig. 3 presents a hollow beam with its dimensions. The size of the beam is modified by changing the parameters p_1 and p_2 of the inside cut out. These parameters are limited by some range $p_{min} < p_i < p_{max}$. The two goals are the minimization of both the mass (g_1) and the deflection (g_2) under the given load F. Fig. 4a plots the position of the best solutions. We find all of them on the boundaries of the parameter range. Fig. 4b indicates that the plot of all the best solutions form a nearly smooth Pareto front.



a) Position of the best solutions in 2D design space b) Pareto front of the beam problem

Fig. 4: Hollow beam problem: positon of best and Pareto front

5 Handling MOO problems using bionic Optimization

Dealing with non-academic MOO problems often requires much computing time and might not yield satisfactory results. Bionic approaches using a parallel search of the solution space and allowing for a representation of the objectives distributions might be more efficient. Hybrid solutions combining the use of meta-models (e.g. Response Surfaces) and bionic search prove to be even better performing in some cases. From a given set of initial studies, we derive meta-models of the different objectives. These meta-models help to identify preliminary positions of Pareto-best solutions and the corresponding values of the goals. Bionic search of the surrounding of the preliminary Pareto front contributes to the improvement of the Pareto front. Switching between the deterministic search on the meta-model and the bionic history yields improved representations of the responses we are looking for.

6 Conclusions

To solve real MOO problems requires large efforts. It is often preferable to reduce them to SOO tasks. If the solution as MOO problems is unavoidable, strategies dealing with the surroundings of the Pareto front, including bionic optimization and meta-models might help to accelerate the process. Especially in the case of irregular responses e.g. due to many local optima, bionic approaches seem to be preferable, as they avoid of sticking to isolated local sub-optima.

References

[1] Wang, Z., Huang, H.-Z., Liu, Y.: A Unified Framework for Integrated Optimization under Uncertainty, *Journal of Mechanical Design* Copyright © 2010 by ASME MAY 2010, Vol. 132.

[2] Gekeler, S., Steinbuch, R. (2014). Remarks on robust and reliable design optimization. Metaheuristics and Engineering, 15th Workshop of the EURO Working Group. S. 77–82.

A Cuckoo Search Algorithm for the flexible ligand-protein docking problem

G. Khensous[1,2] , B. Messabih[2], A. Chouarfia[2]

1. Ecole Normale Supérieure d'Oran (ENS-Oran), BP 1523 El M'Naouer - Route Es-Senia - Oran - Algérie
ghania.khensous@univ-usto.dz

2. Université des Sciences et de la Technologie d'Oran Mohamed Boudiaf, USTO-MB, BP 1505, El M'Naouer, 31000 Oran - Algérie

Keywords : Cuckoo Search Algorithm, Metaheuristic Optimization, Drug Design, Computational Molecular Docking, Flexible Ligand-Protein Docking, Side Chain Flexibility.

1 Abstract

Global optimization is ubiquitous since optimization problems are inherent in nearly every research area, ranging from engineering to the natural sciences such as biology or chemistry. It is also an active research topic in many other areas such as mathematics, business, and the social sciences [1]. Although many efforts have been made to develop global optimization methods, current methods still have difficulties in solving real-world problems with some common but challenging characteristics [2]. As a result, the challenge of developing new methods, baptized Metaheuristics, which are better able to solve difficult problems, still attracts the interests of current researchers. Metaheuristic optimization is therefore a field of growing interest.

Bioinformatics is a multidisciplinary research domain aiming the automatic processing of biological data. One of the Bioinformatics challenges is the CMD "Computational Molecular Docking". CMD is an example of a difficult real-world optimization problem, which consists on the determination of the best possible matching between a protein target and a small organic drug-like compound able to interfere within the biological processes. Therefore, we can consider the docking problem as the problem of finding the global energy minimum of a molecular system corresponding to the best stable structure of the protein-ligand complex. Several optimization methods are used in CMD (see [3-4] for reviews), the most popular ones being GA [5-6] or SA [7-8].

CMD is composed of two steps; the first one is "Searching", the second step is "Scoring". The scoring step allows the evaluation and the ranking of the complex conformations (Ligand-Receptor) found in the searching stage. In the searching step, several simulation methods are used such as systematic search, determinist search and stochastic search [9-10].

In our experiment, we use the Cuckoo Search method which is a stochastic search algorithm inspired by the parasitic breeding behavior of cuckoos [11-12]. We have chosen this algorithm because several studies indicate that Cuckoo Search is a powerful algorithm and successful results have been achieved in various applications such as in manufacturing optimization and physically - based runoff - erosion model [13-14]. A review of the literature, however, reveals no applications in molecular docking optimization and the aim of our work is then to investigate the algorithm's performance to solve the protein-ligand docking problem. On the other hand, the majority of the docking methods use scoring function whereas ours uses an energy function issued from classical molecular mechanics, which includes the polarization energy. In Claverie's force field, the total energy is the sum of four terms: electrostatic, polarization, dispersion, and repulsive energies [15]:

$$E_{tot} = E_{el} + E_{pol} + E_{disp} + E_{rep} \tag{1}$$

A variety of measures exist for evaluating pose predictions [16] but the Root Mean Square Deviation (RMSD) between the experimentally observed heavy atom positions of the ligand and those predicted by the docking program is mostly a standard way to evaluate poses. The RMSD values are calculated using the following expression [17]:

$$x = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\delta_i^2} \qquad (2)$$

N is the atoms number and δ is the distance between atoms' pairs. RMSD unit used in structural biology is the Angstrom (Å) [18].

Therefore, in order to assess the performance of our docking program, we calculated the RMSD between the predicted pose of the ligand and its experimental position. Our docking approach is illustrated on four PDB protein-ligand complexes. RMSD values for all the four cases are less than 2 Å showing the efficiency of our software, and making our docking approaches promising.

References

[1] C. Grebner (2012). New Tabu Search Algorithms for the Exploration of Energy Landscapes of Molecular Systems, WÜRZBURG.

[2] T.T. Nguyen (2006). Incorporating Cultural Adaptation and Local Search into MetaHeuristics Continuous Optimization, Ph.D. dissertation, The University of Birmingham.

[3] S. Y. Huang, X. Zou (2010). Review Advances and Challenges in Protein Ligand Docking, International Journal of Molecular Sciences.

[4] D. B. Kitchen, H. Decornez, J. R. Furr, J. Bajorath (2004). Docking and Scoring in Virtual Screening for Drug Discovery: Methods and Applications, Nature Reviews.

[5] H. M. Alsafi, I. F. Al-Shaikhli (2012). Rational Drug Design using Genetic Algorithm: Case of Malaria Disease, Journal of Emerging Trends in Computing and Information Sciences. Vol. 3, No 7.

[6] S. M. Camila, J. C. B. Hélio E. D. Laurent (2004). A genetic algorithm for the ligand-protein docking problem, Genetics and Molecular Biology, 27, 4, 605-610.

[7] K. Lee, K. Czaplewski, S. Y. Kim, J. Lee (2004). An Efficient Molecular Docking Using Conformational Space Annealing, Korea Institute for Advanced Study.

[8] J. Wang, P. A. Kollman, I. D. Kuntz (1999). Flexible Ligand Docking: A Multistep Strategy Approach, University of California.

[9] Alexandre BEAUTRAIT (2008). Développement et validation de la plateforme de criblage virtuel VSM-G et étude du domaine FAT de la Kinase d'Adhérence Focale FAK, Université Henri Poincaré – Nancy I.

[10] Belkacem SID (2006). Optimisation topologique de structures par algorithmes génétiques, Université de Technologie de Belfort-Montbéliard.

[11] Xin-She Yang and Suash Deb (2009). Cuckoo Search via Lévy Flights, World Congress on Nature & Biologically Inspired Computing (NaBIC 2009).

[12] Yang X.-S. and Deb S. (2010). Engineering Optimisation by Cuckoo Search, Int. J. Mathematical Modelling and Numerical Optimisation, Vol. 1, No. 4, 330–343 (2010).

[13] Anna Syberfeldt and Simon Lidberg (2012). Real-World Simulation-Based Manufacturing Optimization using Cuckoo Search", Proceedings of the Winter Simulation Conference C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A.M. Uhrmacher, eds.

[14] Celso A. G. Santos, Paula K. M. M. Freire and Sudhanshu K. Mishra (2012).Cuckoo Search via Lévy Flights for Optimization of a Physically - Based Runoff - Erosion Model", Journal ofUrban and Environmental Engineering (JUEE), v.6, n.2, p.123-131.

[15] J. Caillet, P. Claverie, Theoretical Evaluation of the Intermolecular Interacting Energy of a Crystal: Application to the Analysis of Crystal Geometry, Acta Cryst. 1975, 431-448.

[16] K. L. Damm-Ganamet, R. D. Smith, J. B. Dunbar, J. J. A. Stuckey and H. A. Carlson (2013). CSAR Benchmark Exercise 2011−2012: Evaluation of Results from Docking and Relative Ranking of Blinded Congeneric Series, J. Chem. Inf. Model., 53, 1853−1870.

[17] Xavier CAVIN, INRIA Lorraine, ARC Docking, Optimisation du docking protéine protéine par le calcul et la visualisation haute performance, Université Henri Poincaré.

[18] Jeffrey S. TAYLOR, Roger M. BURNETT, DARWIN : A Program for Docking Flexible Molecules, The Wistar Institute, Philadelphia, Pennsylvania, 2000

A Parallel Primal Heuristic to solve a Integrated Production Planning and Scheduling Problem

Gustavo Campos Menezes[1], Lucas Teodoro de Lima Santos[2], João Fernando Machry Sarubbi[3] and Geraldo Robson Mateus[4]

[1] Departamento de Eletroeletrônica e Computação, Centro Federal de Educação Tecnológica de Minas Gerais, Contagem, MG, Brazil.
gustavo@cefetmg.br

[2] Departamento de Computação, Centro Federal de Educação Tecnológica de Minas Gerais,Belo Horizonte, MG, Brazil..
lucastls@outlook.com

[3] Departamento de Computação, Centro Federal de Educação Tecnológica de Minas Gerais,Belo Horizonte, MG, Brazil..
joao@decom.cefetmg.br

Departamento de Ciência da Computação, Universidade Federal de Minas Gerais, Belo Horizonte ,MG, Brazil.
mateus@dcc.ufmg.br

Abstract. This paper investigates an optimization problem involving the production planning, product allocation and scheduling of products in the largest bulk port terminal existing in Brazil. The main contributions of this article are related to the use of a parallel approach to solve the integrated problem. The methodology uses a combination of heuristics, column generation and optimization package. The computational experiments (based on real cases) showed that the parallel solution was faster in all tested instantes reaching gains close to 90%.

1 Introduction

In 2015, maritime trade reached a total of 9.8 billion tons, a volume comprising mostly containers and dry and liquid bulk cargo [16]. Of this total, 4.5 billion tons are of dry cargoes (coal, iron ore, grains, bauxite, among other products). According to several works as [12], [15] and [13], a careful and optimized analysis of terminal operations is essential to ensure efficient transport of these loads.

In this work, we solve a real problem involving the production planning, product allocation and scheduling of products in the largest bulk port terminal existing in Brazil. This problem can be modeled as the Integrated Product Flow Planning and Scheduling Problem ($PFPSP$), a two decision-making problem that has critical importance to ensure efficient operation at port terminal facilities. The $PFPSP$ can be defined, in general, as follows: there are a set of supply nodes, where products are available for transportation, storage nodes where the products are stocked and demand nodes or delivery subsystem for shipping products. Specialized equipment with predefined capacities is used to transport the products within the network. An equipment route between nodes has a given capacity and handle one product at a time.

The defined problem can be applied in different scenarios, such as: in the mining industry, bulk ports, agroindustry, among others. In all these cases, the supply nodes are the arrival points of products (iron ore, grains, coal), a storage yard, the place where the products are stored and the point of demand a node where the products are delivered in final destinations. There are several challenges involved in storage yards management.Besides, the solutions obtained by investigating this problem can be applied not only to bulk ports but also in other operations that involve the flow of cargo between supply, stock and demand points. Figure 1 highlights the three major nodes related to this problem.

The offer node represents the arrival or supply of products to the system. The stock node is responsible for temporary storage. This storage can be done in sheds, outdoors (in the case of iron ore) or silos (in the case of grains). Finally, the demand node, which represents the destination of the products. This destination can be the end customer, industries or another cargo terminal. Trucks, conveyor belts, ships, and trains can be used to transport products between these nodes.

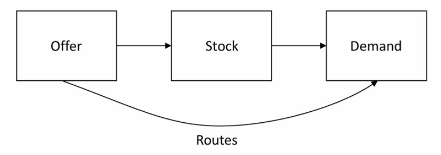


Fig. 1. Offer, Stock and Demand nodes.

In 2017, Menezes et al. [10] proposed an algorithm to solve the $PFPSP$. Their algorithm uses a combination of heuristics and column generation and provides initial solutions to a Branch and Price method. Despite the good results found by Menezes et al. [10] the method had a high computational cost. In some instances, the proposed algorithm spend more than four hours to found a solution. In this research, we proposed a new approach to solve the problem. Our algorithm also uses a combination of heuristics and column generation, but instead of work as a sequential algorithm as Menezes et al.[10], it uses a parallel framework. Our results demonstrate that our parallel algorithm was faster in all tested instantes reaching gains close to 90%.

This paper is organized as follows: section 2 presents the literature review. Section 3 defines the problem and presents some sets and variables used in the mathematical formulation. Section 4 discusses the solution strategy applied. Finally, section 5 is dedicated to computational experiments and the paper ends with conclusions.

2 Literature review

The integrated approach to solve Planning and Scheduling is commonly adopted in the literature. In Grossmann et al. [19], is investigated an integrated production problem, whose goal is to determine at each period which products to manufacture, as well as to establish an optimal capacity modification plan, such that future demand is satisfied. Bruno et al. [5], investigates the integration of Planning and Scheduling of a Network of Batch Plants. The problem is to define the amounts of products to be produced in each time period, the allocation of products to batch units and the detailed timing of operations and sequencing of products. Other researchers in this same direction are the works of: [18], [6], [9], [8]), and, more recently, [11] and [17].

The central problem study in this article involves the flow of products between supply nodes, storage areas, and demand nodes. The references highlighted below are related to mathematical models and algorithms for problems in bulk cargo terminal. Bilgen et al. [2], study the problem of blending and allocating ships for grain transportation. Byung et al., [7] study the allocation of products in the stockyard. This problem is solved using a mixed-integer programming model. Barros et al. [1], develop an integer linear programming model for the problem of allocating berths in conjunction with the storage conditions of the stockyard. Boland [3], address the problem of managing coal stockpiles in Australia. Singh et al. [15], present a mixed-integer programming model for the problem of planning the capacity expansion of the coal production chain in Australia. Finally, Robenek et al. [14], proposes an integrated model for the integrated berth allocation and yard assignment problem in bulk ports.

3 Integrated Production Planning and Scheduling

The mathematical model for the Integrated Production Planning and Scheduling Problem was initially proposed in [10]. All production is planned for a given time horizon, divided into T periods. Routes are classified into three types: routes x that transport products from the Supply node to the Storage yard, routes y from the Supply node to the Demand node, and routes z from the Storage yard to the Demand. The number of routes is limited and they may share equipment. Thus, if two different products are assigned to routes sharing equipment, these routes must be active at non-overlapping intervals. Figure 2 shows a case where two routes (routes 1 and 2) share the same equipment.

The Figure 3 describes at a high level the objective function and the main constraints of the model. In general, the formulation represents the production planning, that is, it defines which

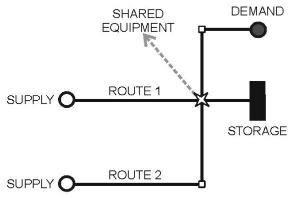
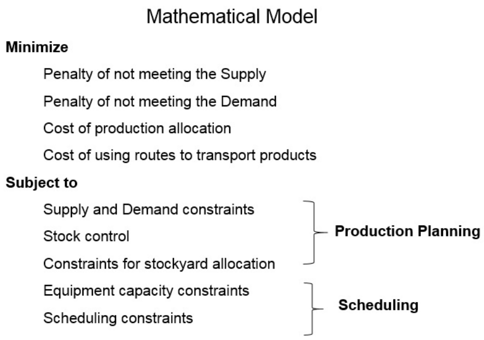


Fig. 2. Routes with shared equipments


Mathematical Model

Minimize

Penalty of not meeting the Supply
Penalty of not meeting the Demand
Cost of production allocation
Cost of using routes to transport products

Subject to

Supply and Demand constraints
Stock control
Constraints for stockyard allocation
Production Planning

Equipment capacity constraints
Scheduling constraints
Scheduling

Fig. 3. PFPSP Model

product, route, the quantity and in what period a product will be transported. The formulation also represents the scheduling problem, that will define the start and the end time of use of each route, guaranteeing that routes sharing equipment are not executed simultaneously.

4 Column Generation method

The PFPSP formulation can be broken into a restricted linear master problem (RMLP), representing the production planning and the pricing subproblem (scheduling), responsible for providing columns for the RMLP. In the PFPSP formulation, the time required to transport the products is limited by the duration of one period. Thus, these constraints determine the start time of each task in a specific period and consider the disjunction between conflicting tasks. These constraints impose great difficulty in solving the PPSFP. An alternative to overcome this obstacle is to divide the period t into micro periods because the time needed to transport the products between offer, stock and demand nodes, are supposed to be smaller than the duration of one period.

To show the entire procedure of solving the PFPSP, we present a small illustration of the method. Suppose that the relaxed PFPSP (by disregarding the scheduling constraints) has been solved and that the following variables (tasks) for the period 3 (three) were extracted from the solution (Table 1). The complete methodology of the column generation procedure can be found in [10].

Table 1. Solution of the relaxed PFPSP (disregarding scheduling constraints)

Vertices	Variables	Values	Routes
A	x^1_{23}	4	R_1
B	x^2_{53}	6	R_2
C	y^8_{553}	3	R_8
D	y^9_{333}	5	R_9
E	z^{10}_{663}	6	R_{10}
F	z^6_{443}	3	R_6

In the first row of Table 1, the variable x^1_{23} (column *Variables*) represents that the product 2 should be carried by Route 1 in period 3, and the total time to transport this product on Route 1 (column *Routes*) will be 4 hours (column *Values*). This variable corresponds to task or vertex A in the conflict graph. The remaining rows of the Table 1 have similar operations. Figure 4 illustrates the conflict between routes. Routes 1 and 2 share equipment and therefore cannot operate simultaneously. The same is true for the routes numbered 6 and 9 and three other routes sharing equipment among themselves (routes 6, 8, and 10).

To generate the weighted conflict graph G (Figure 5), its vertices are defined at column *Vertices* in Table 1, the weight of each vertex is initially considered as the duration (column *Values*) and edges are created considering conflicts of Figure 4.

Once the graph is obtained, the next step is to solve it with one greedy algorithm. Figure 6 presents a possible solution. Based on Figure 6, the number of microperiods obtained (equivalent to

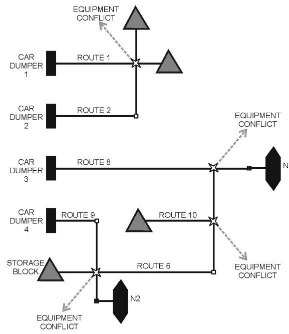


Fig. 4. Routes with conflicts

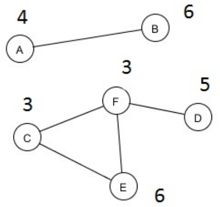


Fig. 5. Weighted conflict graph

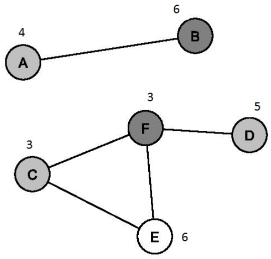


Fig. 6. Heuristic solution

the total number of vertex colors) is three (3). Microperiod one (1) contains the subset of vertices (A, C and D), which are equivalent to variables x^1_{23}, y^8_{553} and y^9_{333}, respectively. The constraints (32), (33), and (34) after the initial column generation are as follows:

$$\begin{array}{rlllll}
x^1_{23} & \leq & \mu_{13} & & & ,\pi^1_{223} \\
x^2_{53} & \leq & & \mu_{23} & & ,\pi^2_{553} \\
y^8_{553} & \leq & \mu_{13} & & & ,\pi^8_{553} \\
y^9_{333} & \leq & \mu_{13} & & & ,\pi^9_{333} \\
z^{10}_{663} & \leq & & & \mu_{33} & ,\pi^{10}_{663} \\
z^6_{443} & \leq & & \mu_{23} & & ,\pi^6_{443}
\end{array}$$

Considering the variables π^1_{223} to π^6_{443}, the values of these dual variables are obtained after the RMLP solution. Now the process of column generation begins: the tasks and values of their dual variables are used to create the weighted conflict graph (Figure 7).

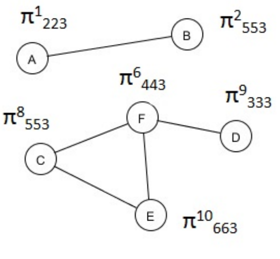


Fig. 7. Weighted graph

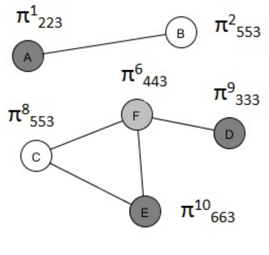


Fig. 8. DSATUR solution

To color a graph G with k colors is equivalent to finding k independent sets. Therefore, a strategy to return more columns to the RMLP in each period is to produce more than one independent set. For this purpose, a heuristic based on the Weighted Vertex Coloring Problem (WVCP) is adopted. From the greedy strategies implemented, the one that has given the best results is DSATUR. It was initially developed for the VCP by [4] based on the classic greedy strategy for this problem. The DSATUR heuristic receives as input the weighted graph (Figure 7) and returns a color for each vertex (Figure 8). For each color of Figure 8, if the solution contains a microperiod of reduced negative cost, the tasks associated with this color will be part of a new microperiod available for the RMLP. The new columns for RMLP are as follows:

$$\begin{array}{rllllllll}
x_{23}^{1} \leq & \mu_{13} & & & & & +\mu_{63} & , \pi_{223}^{1} \\
x_{53}^{2} \leq & & \mu_{23} & & +\mu_{43} & & & , \pi_{553}^{2} \\
y_{553}^{8} \leq & \mu_{13} & & & +\mu_{43} & & & , \pi_{553}^{8} \\
y_{333}^{9} \leq & \mu_{13} & & & & & +\mu_{63} & , \pi_{333}^{9} \\
z_{663}^{10} \leq & & & \mu_{33} & & & +\mu_{63} & , \pi_{663}^{10} \\
z_{443}^{6} \leq & & \mu_{23} & & & +\mu_{53} & & , \pi_{443}^{6}
\end{array}$$

The iterative process continues until the DSATUR heuristic no longer yields attractive columns for the RMLP. At this moment, the optimal solution for the pricing subproblem is found using optimization packages. If this solution contains a column of negative reduced cost, then the iterative process is restarted. Otherwise, the entire column generation procedure is terminated.

4.1 Parallel Approach

At each step that the pricing subproblem is solved, t subproblems are solved (one for each period). Because the subproblems are independent, it is possible to solve them using a parallel approach, which can be applied for both the exact (based on the maximum weight independent set) and heuristic (based on the weighted vertex coloring problem).

The following pseudo-code highlights this procedure. The sequential version of this procedure was initially published in [10]. The code referring to lines 8 through 19 in the procedure detailed in section 4 was parallelized using the parallel programming API (OpenMP (2008)). This tool provides a simple and intuitive API for multithreaded environments. The main characteristics and functionalities can be found in Chapman et al. (2007).

```
 1: procedure ParallelCG                                                      ▷
 2:     $\Omega \leftarrow GetInitialColumns()$                 ▷ generates initial columns for RMLP
 3:     $dual \leftarrow \varnothing$
 4:     $Solution \leftarrow \varnothing$
 5:     repeat
 6:         $dual \leftarrow SolveRMLP(\Omega)$                              ▷ exact solution by solver
 7:         $End \leftarrow true$
 8:         Start Parallel (OpenMp)
 9:         for $i = 1 \rightarrow T$ do                                    ▷ number of periods
10:             $Colors \leftarrow DSATUR(i, dual)$    ▷ Returns only colors with negative reduced cost (WVCP problem)
11:             if $Colors = \varnothing$ then                 ▷ Heuristic above failed to provide new columns
12:                 $Colors \leftarrow ExactSolutionSubproblem(i, dual)$    ▷ exact solution (pricing subproblem)-Cplex parallel mode
13:             end if
14:             if $Colors \neq \varnothing$ then
15:                 $\Omega = \Omega \cup Colors$
16:                 $End = false$
17:             end if
18:         end for
19:          End Parallel
20:     until $End = true$
21:     $Solution \leftarrow SolveRMLP(\Omega)$                          ▷ Cplex parallel mode
22:     $LowerBound = Solution$
23:     if $Fractional(Solution) = True$ then              ▷ checks whether the solution is fractional
24:         $Solution \leftarrow GetIntegerSolution(Solution)$
25:     end if
26:     $UpperBound = Solution$
27: end procedure
```

4.2 Primal Heuristic

Only the conversion of relaxed variables from RMLP to integer,after converging the column generation procedure, and its resolution into a MIP solver, does not provide a good upper bound. To provide a higher quality upper bound, a heuristic was developed for PPSFP, based on a variable-fixing and depth-first search strategy: the fixing and search heuristic (FSH). The following algorithms illustrate the operation of this heuristic.

```
 1: procedure FSHHeuristic                                              ▷
 2:     $Solution$ ← ParallelCG()
 3:     $Value$ ← 0.2
 4:     for $i = 1 \to 3$ do                                              ▷
 5:         if LinearSolution(Solution) then                              ▷
 6:             $FSHFase1(Value)$
 7:             $Solution$ ← ParallelCG()
 8:             $Value = Value + 0.2$
 9:         end if
10:     end for
11:     if ( (IntegerSolution(Solution)) or (Solution = infeasible) ) then
12:         Return Solution
13:     end if
14:     repeat
15:         $FSHFase2()$
16:         $Solution$ ← ParallelCG()
17:     until ( (Solution = Integer) or (Solution = infeasible) )
18:     Return Solution
19: end procedure
```

The FSH Heuristic is divided into two phases: in Phase 1 (lines 4 to 10), consider the optimum solution of the RMLP (when there are no more columns with a reduced negative cost), line 2 of the FSHHeuristic procedure. From the initial node, for each subarea and period, set the most fractional patio allocation variable (f^s_{pt}) to 1, whose quantity of stored product (variable ${}^s_{pt}$ s is greater than 80% of the subarea capacity. Then start the parallel column generation process again: solve each iteration for the RMLP and the pricing subproblem until there are no more columns with negative cost to be inserted. If the solution found is either whole or unfeasible, quit the heuristic. Otherwise, create another node, now setting allocation variables whose inventory is greater than 60% of capacity. Repeat the whole process again. If the solution found is not yet complete, repeat for 40 % of capacity. This heuristic step is described between lines 4 through 10 of the pseudocode. The rate parameter (line 6) corresponds to the values of 80%, 60% and 40% (which are incremented in line 8).

Finally, if an entire solution has not yet been found, the resolution of Phase 2 starts (lines 15 to 18): select the most fractional allocation variable, set its value to 1, and repeat the column generation process (line 16). Repeat Phase 2 until an entire solution or an impractical solution is found.

5 Computational Experiments

The experiments are performed based on a real product flow problem in an iron ore port terminal in Brazil, recognized as one of the largest worldwide. The basic parameters are the number of periods, the products and the routes. In general, they work with seven periods of one day or fourteen periods of twelve hours. The experiments were conducted using a computer cluster, composed of XEON processors with 8 physical threads, hyperthread and 32 GB RAM. The machines are connected by a Gigabit network, running version 12.5 of the CPLEX solver. For all instances, a time limit of 5 hours was set. Table 2 highlights the main parameters used to create the instances.

The first and second columns of Table 3, contains the type, and name of each instance. For the first column (Type), the instances were divided into three levels of difficulty: For the first set (type 1 Instances), it was assumed that any one of the P available products could be allocated to any subarea. For the second set (type 2), the full set of products is split into two subsets: the first subset

Table 2. Data used to generate the instances

Parameter	Description
Stockyard	The product storage area is divided into four stockyards.
Delivery	Two berths: two ships can be loaded simultaneously at berth 1.
Equipment	Five car dumpers, four ore reclaimers, three stackers/reclaimers (equipment that performs both tasks), and eight stackers.
α_{pt}	2 (two monetary units)
β_{npt}	10 (ten monetary units for the berth number one), 50 (fifty monetary units for the berth number two).
$\gamma^{s}_{p,p',t}$	10 (ten monetary units)
$\lambda_{pp'}$	Based on the following formula: 0.01 (monetary unit) $*\|p-p'\|$, where $\|p-p'\|$ represents the quality deviation between the product p and p'.
σ^{r}	Based on the following formula: 0.01 (monetary unit) $*$ length of route r.

of products could be stored in any subarea of stockyards 1 and 2, and the second subset could be stored in the subareas of stockyards 3 and 4. Finally, for the third set (type 3), the products were split into four subsets, each one allocated to one of the four stockyards. The instance $8P5Prod$ corresponds a planning horizon of 8 periods, and 5 different products being handled. Columns Z_{LB} and Z_{UB} provide the best lower and upper bounds, respectively. The GAP column provides the solution, computed as, $GAP = 100(Z_{UB} - Z_{LB})/Z_{UB}$, t_s are the elapsed computational times, expressed in seconds. The character $(-)$ represents instances for which the solver could not obtain a solution for the PFPSP due to insufficient memory.

The Table 3 presents three solution approaches for PFPSP: sequential heuristic, parallel heuristic and exact solution (through the use of the CPLEX solver). The results shown in Table 3 indicate that solving the PFPSP in optimization packages is not feasible. The solver was able to produce solutions only for 20 instances. In the rest, because insufficient memory, it was not possible to obtain even an upper bound. With the heuristic, it is possible to obtain solutions for all instances, all supplies and demands were met. As expected, the parallel heuristic solved instances in less time. Its performance was superior to sequential heuristics and to the CPLEX solver in all cases.

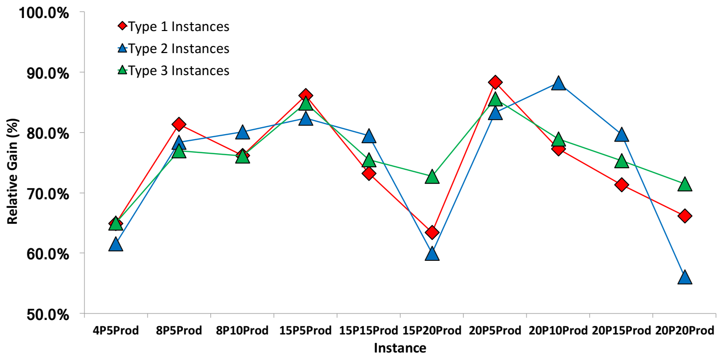


Fig. 9. Relative gain between parallel and sequential solution

The Figure 9 represents the relative gain, in terms of time, between parallel and sequential solution. The x-axis represents several instances while the y-axis represents the relative gain in percentage. When a gain reach 60% means that the parallel algorithm was 60% faster than the sequential approach.

6 Final Remarks

In this work, we consider an integrated problem of planning and scheduling. The problem is general and can be applied to represent various scenarios related to the flow of bulk cargo (iron ore, coal, and grains). It was proposed a Parallel approach to solve the PFPSP. Computational experiments

Table 3. Computational Experiments

		Sequential Heuristic				**Parallel Heuristic**		CPLEX			
Type	Instance	Z_{LB}	Z_{UB}	t(s)	Gap(%)	**t(s)**	**Number of nodes**	Z_{LB}	Z_{UB}	t(s)	Gap(%)
1	4P5Prod	67.55	68.40	57	1.24	**20**	**9**	68.27	68.27	219	0.00
	4P10Prod	77.32	91.21	518	15.22	**175**	**15**	82.23	82.23	3354	0.00
	8P5Prod	125.69	150.03	241	16.22	**45**	**21**	128.61	128.61	1241	0.00
	8P10Prod	141.18	259.18	1410	45.53	**336**	**24**	-	259.18	1410	-
	15P5Prod	234.13	292.38	375	19.92	**52**	**13**	243.62	243.64	16664	0.00
	15P10Prod	298.25	547.85	4770	45.56	**1069**	**39**	301.42	922432.00	18007	0.99
	15P15Prod	171266.00	171392.00	6820	0.07	**1824**	**38**	-	171392.00	6820	-
	15P20Prod	433229.00	433251.00	2291	0.01	**838**	**17**	-	433251.00	2291	-
	20P5Prod	331.74	391.16	1330	15.19	**155**	**43**	341.81	465.94	1330	0.26
	20P10Prod	480.14	644.34	5617	25.48	**1275**	**40**	-	644.34	5617	-
	20P15Prod	211939.00	212026.00	7339	0.04	**2104**	**32**	-	212026.00	7339	-
	20P20Prod	664364.00	664416.00	4026	0.01	**1363**	**19**	-	664416.00	4026	-
2	4P5Prod	77.94	78.08	13	0.18	**5**	**9**	78.01	78.01	69	0.00
	4P10Prod	100.59	111.44	96	9.74	**34**	**16**	103.29	103.29	255	0.00
	8P5Prod	146.98	147.11	37	0.09	**8**	**14**	147.11	147.11	86	0.00
	8P10Prod	170.41	244.52	236	30.31	**47**	**19**	175.84	175.84	433	0.00
	15P5Prod	274.54	275.13	68	0.21	**12**	**14**	275.13	275.13	622	0.00
	15P10Prod	350.31	394.14	358	11.12	**49**	**17**	362.38	362.39	3950	0.00
	15P15Prod	509.26	629.03	2024	19.04	**415**	**42**	-			-
	15P20Prod	587.68	731.79	6026	19.69	**2410**	**42**	-			-
	20P5Prod	382.62	383.83	96	0.32	**16**	**14**	383.83	383.83	858	0.00
	20P10Prod	553.18	633.40	700	12.66	**82**	**23**	570.5	570.50	6203	0.00
	20P15Prod	703.22	932.54	2884	24.59	**584**	**41**	738.50	42106.40	18006	0.98
	20P20Prod	761.28	1022.87	11922	25.57	**5241**	**68**	-			-
3	4P5Prod	73.45	73.72	20	0.36	**7**	**10**	73.72	73.72	32	0.00
	8P5Prod	137.03	149.06	87	8.08	**20**	**15**	137.75	137.75	230	0.00
	8P10Prod	155.76	220.13	414	29.24	**99**	**19**	160675.00	160.68	1856	0.00
	15P5Prod	254.53	280.16	212	9.15	**32**	**19**	256819.00	256.82	1157	0.00
	15P15Prod	461.89	605.08	4715	23.66	**1154**	**57**	-			-
	15P20Prod	555.41	914.35	11847	39.26	**3228**	**76**	-			-
	20P5Prod	357.78	384.69	299	7.00	**43**	**26**	361281.00	361.28	2101	0.00
	20P10Prod	507.79	617.94	2145	17.83	**452**	**39**	-			-
	20P15Prod	641.92	860.85	7376	25.43	**1820**	**63**	-			-
	20P20Prod	714.18	1159.96	17058	38.43	**4867**	**76**	-			-

were conducted considering a sequential and parallel version of the method, and the results were compared to an exact approach based on CPLEX solver. As a future work, a parallel branch and price algorithm is being developed. Another research topic is the development of a stochastic optimization model, capable of considering uncertainty surrounding supply, demand and equipment failure.

Acknowledgements
This research is supported by the following institutions: Fundação de Amparo à Pesquisa do Estado de Minas Gerais (FAPEMIG) and Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq).

References

1. V. H. Barros, T. S. Costa, A. C. M. Oliveira, and L. A. N. Lorena. Model and heuristic for berth allocation in tidal bulk ports with stock level constraints. *Computers & Industrial Engineering*, 60:606 – 613, 2011.
2. B. Bilgen and I. Ozkarahan. A mixed-integer linear programming model for bulk grain blending and shipping. *International Journal of Production Economics*, 107(2):555 – 571, 2007.
3. N. Boland, D. Gulezynski, and M. Savelsbergh. A stockyard planning problem. *EURO Journal on Transportation and Logistics*, 1(3):197–236, 2012.
4. D. Brélaz. New methods to color the vertices of a graph. *Commun. ACM*, 22(4):251–256, Apr. 1979.
5. B. A. Calfa, A. Agarwal, I. E. Grossmann, and J. M. Wassick. Hybrid bilevel-lagrangean decomposition scheme for the integration of planning and scheduling of a network of batch plants. *Industrial & Engineering Chemistry Research*, 52(5):2152–2167, 2013.
6. F. Gaglioppa, L. A. Miller, and S. Benjaafar. Multitask and multistage production planning and scheduling for process industries. *Operations Research*, 56(4):1010–1025, 2008.

7. B. Kim, B. Koo, and B. S. Park. A raw material storage yard allocation problem for a large-scale steelworks. *The International Journal of Advanced Manufacturing Technology*, 41(9-10):880–884, 2009.
8. T. Kis and A. Kovcs. A cutting plane approach for integrated planning and scheduling. *Computers and Operations Research*, 39(2):320–327, 2012.
9. G. R. Mateus, M. G. Ravetti, M. C. Souza, and T. M. Valeriano. Capacitated lot sizing and sequence dependent setup scheduling: an iterative approach for integration. *Journal of Scheduling*, 13(3):245–259, 2010.
10. G. C. Menezes, G. R. Mateus, and M. G. Ravetti. A branch and price algorithm to solve the integrated production planning and scheduling in bulk ports. *European Journal of Operational Research*, 258(3):926 – 937, 2017.
11. H. Meyr and M. Mann. A decomposition approach for the general lotsizing and scheduling problem for parallel production lines. *European Journal of Operational Research*, 229(3):718 – 731, 2013.
12. A. M. Newman, R. Rubio, R. Caro, A. Weintraub, and K. Eurek. A review of operations research in mine planning. *Interfaces*, 40(3):222–245, 2010.
13. B. S. Pimentel, G. R. Mateus, and F. A. Almeida. Stochastic capacity planning and dynamic network design. *International Journal of Production Economics*, 145(1):139 – 149, 2013.
14. T. Robenek, N. Umang, and M. Bierlaire. A branch-and-price algorithm to solve the integrated berth allocation and yard assignment problem in bulk ports. *European Journal of Operational Research*, 235(2):399 – 411, 2014.
15. G. Singh, D. Sier, A. T. Ernst, O. Gavriliouk, R. Oyston, T. Giles, and P. Welgama. A mixed integer programming model for long term capacity expansion planning: A case study from the hunter valley coal chain. *European Journal of Operational Research*, 220(1):210 – 224, 2012.
16. UNCTAD. Review of maritime transport. Technical report, United Nations Conference on Trade and Development, http://www.unctad.org, 2016.
17. C. Wolosewicz, S. Dauzre-Prs, and R. Aggoune. A Lagrangian heuristic for an integrated lot-sizing and fixed scheduling problem. *European Journal of Operational Research*, 244(1):3–12, 2015.
18. D. Wu and M. Ierapetritou. Hierarchical approach for production planning and scheduling under uncertainty. *Chemical Engineering and Processing: Process Intensification*, 46(11):1129 – 1140, 2007.
19. F. You, I. E. Grossmann, and J. M. Wassick. Multisite capacity, production, and distribution planning with reactor modifications: Milp model, bilevel decomposition algorithm versus lagrangean decomposition scheme. *Industrial & Engineering Chemistry Research*, 50(9):4831–4849, 2011.

WPBO: A New Metaheuristic Technique Inspired from Wolf Pack Behaviour

Abida TOUMI[1], Abir BETKA[1], Salim SBAA[1], and Nadjiba TERKI[1]

Electrical Engineering Department, Faculty of Technology,
University of Biskra, 07000 Biskra, Algeria
abida_ba@yahoo.fr
abir_betka@hotmail.fr
s_sbaa@yahoo.fr
t_nadjiba@yahoo.fr

Abstract

In recent years, there is a significant direction of researchers toward metaheuristics to solve hard optimization problems. All metaheuristic techniques are inspired from natural phenomena and animals' social behaviours. In this paper a new metaheuristic optimization technique inspired from Wolf Pack Behaviour is proposed and presented. This algorithm imitate the special interaction existing between different members of a specified wolfs pack. To evaluate the WPBO performances, the presented algorithm is benchmarked on 23 well-known test functions and compared with different metaheuristic techniques. the performnces of our algorithms are also tested in high dimensions, six test function contain 150, 200 and 500 variables are used. The comparison results show that the proposed algorithm outperforms other algorithms in many benchmark functions.

Keyword

Metaheuristics, Swarm Intelligence, Optimization, Wolf Pack Behaviour, WPBO.

1 Introduction

In the last few decades, there has been a growing interest in metaheuristics to solve hard optimization problems, especially when the exact methods are unable to solve these problems or they demand an excessive computational time. Numerous metaheuristic techniques have been proposed and developed in the literature; almost all metaheuristics have some common points; they are inspired from natural phenomena behaviours, they didn't use the gradient of fitness function, they start with a number of candidate solutions generated randomly from the search space then they applied different tools to update the positions of the candidate solutions and converge iteratively to the global optimum solution.

The success of any metaheuristic technique depend to its capacity to ensure a balance between the exploration and exploitation phases of the search space, in order to check different locations and exploit the neighbouring locations of good solutions in the search space [1].

Wolves behaviour has been a source of inspiration of a metaheuristic technique called Grey Wolf Optimizer (GWO) proposed by Mirjalili in 2014 [2], that mimics the hierarchy dominance and hunting mechanism of grey wolves.

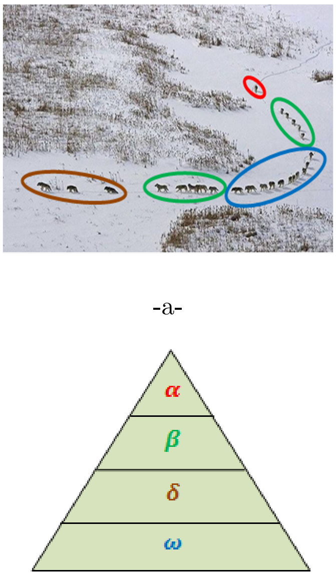
In this paper a new metaheuristic technique inspired from the wolf pack behaviour (WPBO) is proposed, the presented algorithm is so different from the GWO; the initialized population is divided into four groups, based on the fitness values of solutions, each group is subjected to the next higher group and order the next lower group, the number of wolves in each group changed iteratively in order to allow the algorithm to explore efficiently the search space then transit it to the exploitation phase.

The rest of this paper is organized as follow, in the second section the proposed algorithm is explained, the results of the evaluation are discussed in the third section and finally section four concludes with a summary.

2 Wolf Pack behavior

2.1 Inspiration

Wolves are highly social animals that live in packs (Fig.1.a). A pack is an extended family group follows a very strict leading system that can be represented by a pyramid as in figure (Fig.1.b). At the top, there is the leader Alpha, the dominant member, who issues all decisions about eating sleeping and the walking direction...etc. After him there are the subordinate betas, they are the best individuals after alpha; they subject to alpha decisions and order the rest wolves in lower level. In the third level, we find the elders, the guards and the scouts; they are responsible for protection and warning the pack in case of an imminent danger, also, they are the first to be sacrificed in case of ambush. At the pyramid base, we find the rest individuals of the pack that follow the orders of the dominant wolves. [2]



-a-

-b-

Fig. 1: The hierarchical organization of the wolves in the pack

Wolves behaviour has been a source of inspiration of a new metaheuristic technique called Grey Wolf Optimizer (GWO) in 2014 [2], in which the author have modelled mathematically the hunting strategy and the social hierarchy of grey wolves in order to design GWO.
In this algorithm; a number of candidate solutions are initialized randomly in the search space, after the fitness evaluation of each solution, the best solution X_α, the second best solution X_β and the third best solution X_δ are defined then the positions of each solution are updated using (Eq.1)

$$\boldsymbol{X_{t+1}} = \frac{\boldsymbol{X_1} + \boldsymbol{X_2} + \boldsymbol{X_3}}{3} \tag{1}$$

Where;
$\boldsymbol{X_1} = \boldsymbol{X_\alpha} - \boldsymbol{A_1 D_\alpha}$, $\boldsymbol{X_2} = \boldsymbol{X_\beta} - \boldsymbol{A_2 D_\beta}$, $\boldsymbol{X_3} = \boldsymbol{X_\delta} - \boldsymbol{A_3 D_\delta}$

$\boldsymbol{D_\alpha} = |\boldsymbol{C_1 X_\alpha} - \boldsymbol{X_t}|$, $\boldsymbol{D_\beta} = |\boldsymbol{C_2 X_\beta} - \boldsymbol{X_t}|$, $\boldsymbol{D_\delta} = |C_3 \boldsymbol{X_\delta} - \boldsymbol{X_t}|$
$\boldsymbol{A} = 2.\boldsymbol{a}.\boldsymbol{r_1} - \boldsymbol{a}$
$\boldsymbol{C} = 2.\boldsymbol{r_2}$
$\boldsymbol{r_1}$ and $\boldsymbol{r_2}$ are two random vectors in [0,1]
To transit the algorithm from exploration to exploitation $\boldsymbol{a}$ takes a decreasing value from 2 to 0. Like all the metaheuristics, the GWOs processes are repeated until the maximum number of iteration is achieved.
In this paper, we have proposed a new metaheuristic technique inspired also from wolf behaviour, it shares with GWO the same inspiration but the algorithm is different.

2.2 Wolf pack behaviour optimization

While Wolf pack behaviour optimization (WPBO) is a population-based metaheuristic technique, we start the algorithm by a random initialization of a number N_p of candidate solutions, after that in order to model the hierarchical organization of the wolves in the pack, the candidate solutions are ranked from the best solution to the worst then we divide the population into four groups; G_α, G_β, G_δ and G_ω representing the first, the second, the third and the fourth group respectively. The difference between WPBO and GWO is that in WPBO α, β and δ are groups of solutions and not one solution for each one.

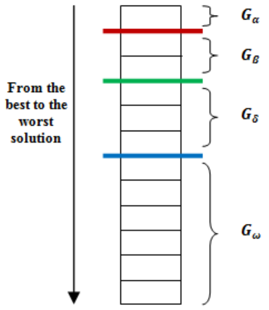


Fig. 2: The population partition

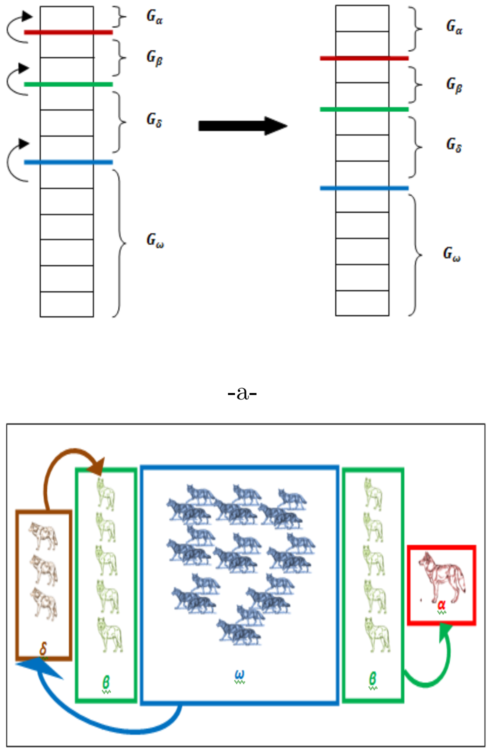
As mentioned before, the wolves in each level are subjected to the orders of the wolves in the adjacent higher level and in their part; they order the wolves in the next lower level. To model mathematically the dominance discipline of wolves, each candidate solution updates its positions by using the following equations;

$$\boldsymbol{X_{t+1}} = \boldsymbol{X_g} - \boldsymbol{A_1}.\boldsymbol{D_g} \tag{2}$$

$$\boldsymbol{D_g} = \boldsymbol{C_1}.\boldsymbol{X_g} - \boldsymbol{X_t} \tag{3}$$

If the candidate solution belongs to the first group G_α; $\boldsymbol{X_g}$ represents the global best solution found until the current iteration, if it belongs to the second group G_β; $\boldsymbol{X_g}$ represents a candidate solution randomly selected from G_α, if it belongs to the third group G_δ; $\boldsymbol{X_g}$ represents a candidate solution randomly selected from G_β but if it belongs to the fourth group G_ω ; $\boldsymbol{X_g}$ represents a candidate solution randomly selected from G_δ. The difference in $\boldsymbol{X_g}$ between the solutions allows the algorithm to explore the search space better.
The two vectors $\boldsymbol{A_1}$ and $\boldsymbol{C_1}$ are defined like in GWO algorithm. To transit the algorithm from exploration to exploitation, we have proposed to move one solution from a level to the upper level at each iteration, with this manner in the last iteration all solution belong to G_α and they are attracted toward the global best solution.

If the strategy of passing the solutions from one level to the other begin since $iter = \frac{Maximum\,iteration}{2}$ the exploration will be improved, but if it starts since the first iteration $iter = 1$ the exploitation will



-a-

-b-

Fig. 3: The strategy of solutions moving from a level to another

be improved, for that we have proposed two versions of WPBO; WPBO1 ($iter = \frac{Maximum\,iteration}{2}$) and WPBO2 ($iter = 1$).
The proposed algorithm can be presented in Algorithm 1;

Algorithm 1 WPBO algorithm

```
Initialize the parameters ($N_p$ and $Iter_{max}$)
Random initialization of candidate solutions
for $t = 1 : Iter_{max}$ do
    Update the parameters A, a and C
    Update the groups
    Update the positions of candidate solutions
    Evaluation of candidate solutions with the fitness function
End for
```

3 Experimental results

The WPBO performances are tested in two phases. Firstly the presented algorithms are benchmarked on 23 well-known test functions. Then the performances of our algorithms are analysed in high dimensions test functions.

3.1 Experiment 1

In the first part, the presented algorithms are benchmarked on 23 well-known test functions described in Appendix A and compared with different metaheuristic techniques such as Particle Swarm Optimization (PSO) [3], Artificial Bee Colony (ABC) [4], Cultural Algorithm (CA) [5], Sine Cosine Algorithm (SCA) [6] and Grey Wolf Optimizer (GWO) [2].

The curves of fitness values according to the iteration number obtained with F_{01}, F_{03}, F_{11} and F_{22} functions are illustrated in figures (Fig4-Fig7), a statistical test with ANOVA (analysis of variance) method is provided in figure (Fig.8)

Table 1: Optimization results of our algorithm and other metaheuristics

Functions		Optimum	PSO	ABC	CA	SCA	GWO	WPBO 1	WPBO 2
F_{01}	Mean	0	0.3970	0.1336	1.2946e-04	1.2728	8.1334e-41	1.6662e-221	3.7581e-238
	StDev		0.3209	0.0324	1.3350e-04	3.3294	1.1758e-40	0	0
	Rank		6	5	4	7	3	2	1
F_{02}	Mean	0	0.0352	0.1752	59.2268	2.3977e-06	8.0399e-47	7.7689e-220	2.5781e-229
	StDev		0.1624	0.0990	20.9776	6.6258e-06	8.3453e-47	0	0
	Rank		5	6	7	4	3	2	1
F_{03}	Mean	0	1.6452e+003	8.4519e+03	2.6431e+04	3.8001e+03	2.8663e-11	9.6296e-215	5.9934e-233
	StDev		640.9846	1.7992e+03	4.6212e+03	3.0736e+03	7.8706e-11	0	0
	Rank		4	6	7	5	3	2	1
F_{04}	Mean	0	2.1999	12.3781	3.1395	5.3804	1.3831e-21	2.1367e-230	3.4521e-239
	StDev		1.7124	1.6148	3.2499	5.9093	2.8568e-21	0	0
	Rank		4	7	5	6	3	2	1
F_{05}	Mean	0	33.0197	27.8865	48.5678	27.0538	25.8985	28.8868	28.8480
	StDev		31.9313	0.4019	115.9528	0.7684	0.7736	0.0366	0.0526
	Rank		6	3	7	2	1	5	4
F_{06}	Mean	0	2.7682e+004	3.3316e+04	1.5645e+04	3.1148e+04	1.2398e+03	6.1978	6.1636
	StDev		4.9476e+003	5.1635e+03	3.3031e+03	5.9109e+03	330.2118	1.0394	1.0077
	Rank		5	7	4	6	3	2	1
F_{07}	Mean	0	0.0251	0.0311	0.0701	0.0086	1.9404e-04	3.0110e-05	1.1959e-05
	StDev		0.0170	0.0079	0.0177	0.0085	9.6654e-05	2.8478e-05	1.1354e-05
	Rank		5	6	7	4	3	2	1
F_{08}	Mean	-418.9829 * n	-9.9460e+003	-9.1745e+32	-1.0400e+04	-4.3341e+03	-6.3427e+03	-6.8723e+03	-6.7581e+03
	StDev		514.9218	2.1229e+33	669.8460	285.6626	818.9499	1.0918e+03	1.0281e+03
	Rank		3	1	2	7	6	4	5

Functions		Optimum	PSO	ABC	CA	SCA	GWO	WPBO 1	WPBO 2
F_{09}	Mean	0	275.2287	308.3371	316.9494	231.5441	47.6089	6.1550e-12	1.5007e-11
	StDev		17.2402	20.9358	18.6367	47.0853	15.1017	2.2175e-11	4.2357e-11
	Rank		5	6	7	4	3	1	2
F_{10}	Mean	8.8818e-16	13.4428	16.4168	11.1855	16.7991	0.0844	9.2145e-12	1.7497e-12
	StDev		0.7387	0.7822	0.8800	3.8739	0.0292	1.7279e-11	4.8074e-12
	Rank		5	6	4	7	3	1	2
F_{11}	Mean	0	43.5010	57.5828	18.1817	36.2411	0.0848	0	0
	StDev		11.9431	9.5749	4.6308	15.6611	0.0495	0	0
	Rank		5	6	3	4	2	1	1
F_{12}	Mean	1.5705e-032	0.0456	0.5025	4.1989	0.3651	0.0120	0.1877	0.2152
	StDev		0.0902	0.1092	5.9838	0.0549	0.0119	0.0709	0.1058
	Rank		2	6	7	5	1	3	4
F_{13}	Mean	0	0.0040	0.1013	0.0062	2.0343	0.2046	2.2251	2.2097
	StDev		0.0095	0.0132	0.0096	0.1694	0.1377	0.5291	0.4361
	Rank		1	3	2	5	4	7	6
F_{14}	Mean	0.998004	0.9980	1.0297	1.1561	1.7920	3.3878	1.5121	1.5529
	StDev		4.3578e-013	0.1571	0.7905	0.9918	3.8337	1.3110	1.0721
	Rank		1	2	3	6	7	4	5
F_{15}	Mean	0.0003075	0.0036	0.0010	0.0044	9.5601e-04	0.0019	5.4946e-04	7.2135e-04
	StDev		0.0075	3.3967e-05	0.0122	3.9461e-04	0.0056	3.7901e-04	5.1893e-04
	Rank		6	4	7	3	5	1	2
F_{16}	Mean	-1.0316285	-1.0316	-1.0315	-1.0316	-1.0316	-1.0316	-1.0316	-1.0316
	StDev		7.0682e-013	7.6299e-05	6.2476e-16	5.2826e-05	1.6968e-08	1.5051e-05	4.5242e-05
	Rank		2	7	1	6	3	4	5
F_{17}	Mean	0.398	0.3979	0.3980	0.3979	0.3999	0.3979	0.3979	0.3979
	StDev		1.0120e-011	6.9784e-05	0	0.0017	1.5976e-06	9.4208e-05	4.9353e-05
	Rank		3	1	2	7	4	6	5
F_{18}	Mean	3	3.0000	3.0001	3.0000	3.0000	3.0000	3.0004	3.0003
	StDev		1.8965e-012	1.8513e-04	1.3628e-15	9.3354e-05	2.1140e-05	6.6328e-04	3.6212e-04
	Rank		2	5	1	4	3	7	6
F_{19}	Mean	-3.86	-3.8628	-3.8625	-3.8628	-3.8537	-3.8612	-3.8184	-3.8304
	StDev		1.0583e-007	2.3838e-04	1.8467e-15	0.0037	0.0025	0.0748	0.0659
	Rank		3	4	2	5	1	7	6
F_{20}	Mean	-3.32	-3.2602	-3.2515	-3.2792	-3.0135	-3.2608	-3.1685	-3.1762
	StDev		0.0606	0.0403	0.0582	0.2057	0.0732	0.0921	0.1083
	Rank		3	4	1	7	2	6	5
F_{21}	Mean	-10.1532	-6.9258	-10.1018	-7.7627	-3.2472	-9.9496	-10.1500	-10.1326
	StDev		3.0358	0.1039	3.5566	1.6226	1.0105	0.0038	0.0241
	Rank		6	3	5	7	4	1	2
F_{22}	Mean	-10.4029	-7.8961	-10.3587	-7.5721	-4.5168	-10.4009	-10.3945	-10.3620
	StDev		3.4918	0.0345	3.5908	1.4180	0.0011	0.0098	0.0961
	Rank		5	4	6	7	1	2	3
F_{23}	Mean	-10.5364	-8.5370	-10.4930	-7.7253	-4.4472	-10.3185	-10.5300	-10.5180
	StDev		3.3054	0.0230	3.6025	0.9653	1.0813	0.0111	0.0342
	Rank		5	3	6	7	4	1	2
Overall Rank			4	6	5	7	2	3	1

The difference between the two proposed algorithms, WPBO1 and WPBO2, is in the number of iterations from which the algorithm begins to move the solutions from one level to the other and change the sizes of the groups. And since this strategy makes the solutions more and more attracted to the global best solution, so we can say that the difference between WPBO1 and WPBO2 algorithms lies in the exploration and exploitation phases of each one of them.

In WPBO1 algorithm, the level changing of solutions starts at $iter = \frac{Maximum\,iteration}{2}$, which gives to the algorithm more time to explore the search space, from the comparison results in table 1, WPBO1 surpass WPBO2 in multimodal functions, especially F_{21}, F_{22} and F_{23}, whose need more exploration to avoid the attraction to local optimums. Unlike, WPBO2 that starts from the first iteration, the solution move to the first group G_α, faster, which allows the algorithm to concentrate the search in potential solution and improve the exploitation by consequence. The results in F_1, F_2, F_3 and F_4, validate this concept.

The influence of large exploration time given by WPBO1 is clearly presented by the evolution of fitness functions in figures from Fig.4 to Fig.7, and the ANOVA test of some chosen functions in Fig.8. Also, the rapid move to the exploitation phase proposed in WPBO2 has proved its benefits by the improvement of the convergence to the optimal solution, and this is shown in figures from Fig.4 to Fig.7 too.

These results show that our proposed algorithm has, in most cases, outperformed the other algorithms

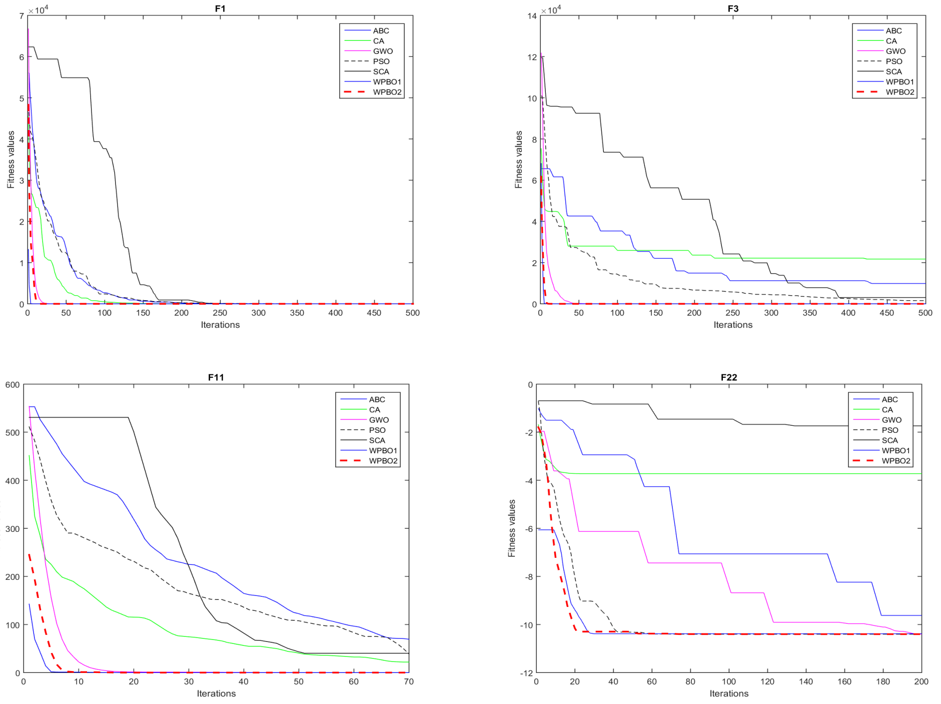


Fig. 4: Fitness curves of F_{01}, F_{03}, F_{11} and F_{22}

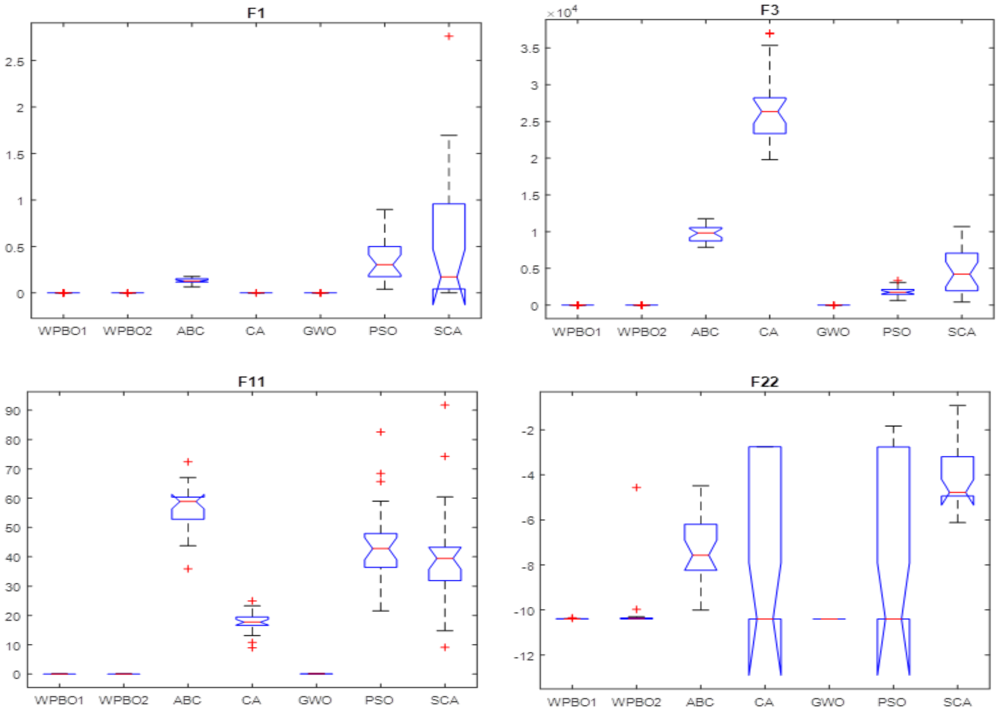


Fig. 5: ANOVA test of F_{01}, F_{03}, F_{11} and F_{22}

3.2 Experiment 2

To evaluate the performances of our algorithms in high dimensional functions. Six test functions selected from the precedent functions, contain 150, 200 and 500 variables, are used. the results of our algorithms are compared with different swarm intelligence algorithms like PSO, ABC and GWO. Starting from the results represent the tables 2,3 and 4, our algorithms WPBO1 and WPBO2 outperform all the other swarm itelligence algorithms in all the used test functions. Which prove their effectivenss in high dimensional functions.

Table 2: Optimization results with $dim = 150$

Functions	PSO	ABC	GWO	WPBO 1	WPBO 2
F1	27883,92	68567,97	1,35E-14	4,50E-228	7,38E-244
F2	101,07	2,52E+40	1,65E-18	5,15E-226	9,62E-239
F4	45,29	96,94	0,00215	6,30E-236	2,21E-249
F6	339866,52	401426,36	74698,36	34,3225	37,1750
F9	2010,11	2375,02	722,825	1,16E-11	7,73E-12
F11	1746,50	2843,67	6,2109	0	0

Table 3: Optimization results with $dim = 200$

Functions	PSO	ABC	GWO	WPBO 1	WPBO 2
F1	52954,56	200068,67	1,69E-11	4,60E-229	1,53E-237
F2	219,7935	1,61E+62	7,47E-16	3,31E-221	2,16E-235
F4	49,6474	97,7190	1,7195	2,88E-231	8,72E-248
F6	385834,30	581699,86	105657,38	48,28	49,48
F9	2740,51	3359,65	1263,94	9,09E-13	2,32E-11
F11	2909,58	4629,14	16,0304	0	0

Table 4: Optimization results with $dim = 500$

Functions	PSO	ABC	GWO	WPBO 1	WPBO 2
F1	404901,68	1490066,96	1,15E-05	2,52E-223	8,07E-234
F2	1416,61	6,32E+220	1,01E-09	1,92E-216	1,84E-231
F4	62,6579	99,3180	43,2951	1,34E-237	6,20E-245
F6	1208082,56	1486091,90	511209,11	127,5441	123,2278
F9	7585,52	8664,04	4356,54	9,09E-13	9,09E-13
F11	8046,87	13376,62	319,37	0	0

4 Conclusion

A new metaheuristic inspired from the relation between different members of a wolf pack was presented and evaluated through diverse benchmark functions. Two versions of the WPBO algorithm were suggested, one improves the exploration ability, and the other improves the exploitation, and they both have given excellent results comparing with other well known metaheuristics.

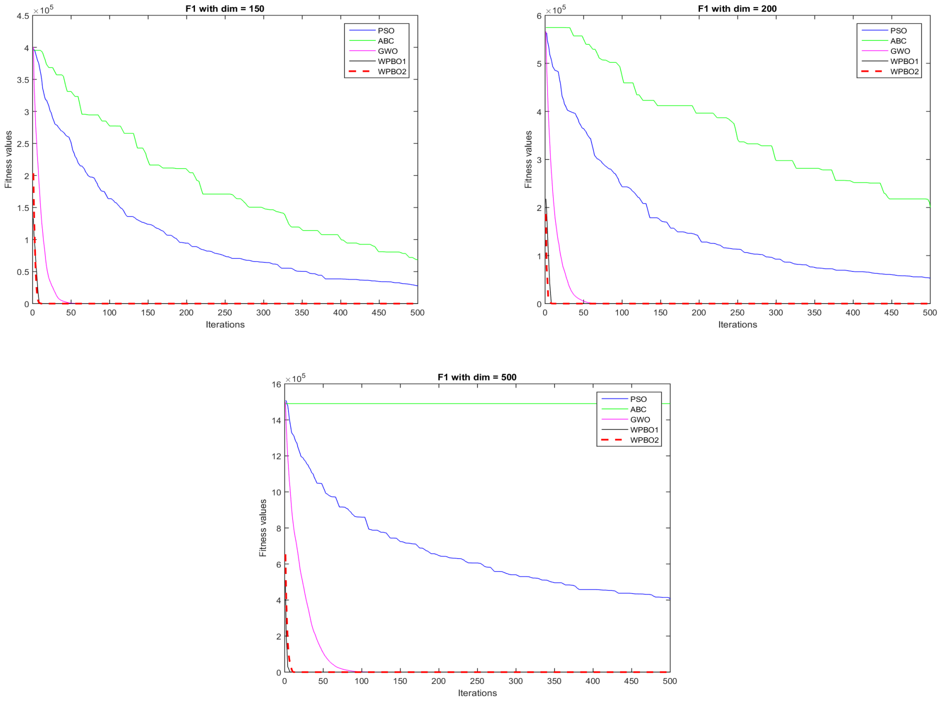


Fig. 6: Fitness curves of F_{01} with high dimensions

Appendix A

Test Functions	**Dimension**	**Range**	**Maximum Iteration**
$F_{01} = \sum_{i=1}^{n} x_i^2$	30	[-100,+100]	500
$F_{02} = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	[-10,+10]	950
$F_{03} = \sum_{i=1}^{n} (\sum_{j=1}^{i} x_j)^2$	30	[-100,+100]	500
$F_{04} = \max_i \{\lvert x_i \rvert, 1 \leq i \leq D\}$	30	[-100,+100]	1000
$F_{05} = \sum_{i=1}^{D-1} [100 \times (x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	30	[-30,+30]	8000
$F_{06} = \sum_{i=1}^{D} ([x_i + 0.5])^2$	30	[-100,+100]	15
$F_{07} = \sum_{i=1}^{D} [ix_i^4 + random[0,1)]$	30	[-1.28,+1.28]	1500
$F_{08} = \sum_{i=1}^{D} -x_i \sin(\sqrt{\lvert x_i \rvert})$	30	[-500,+500]	1500
$F_{09} = \sum_{i=1}^{D} [x_i^2 - 10\cos(2\pi x_i) + 10]$	30	[-5.12,+5.12]	40
$F_{10} = -20exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2})$ $-exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)) + 20 + e$	30	[-32,+32]	60
$F_{11} = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D}\cos(\frac{x_i}{\sqrt{i}}) + 1$	30	[-600,+600]	70
$F_{12} = \quad \frac{\pi}{D}\{10\sin^2(\pi y_i)$ $+\sum_{i=1}^{D-1}(y_i - 1)^2\,[1 + 10\sin^2(\pi y_i + 1)]$ $+(yD - 1)^2 + \sum_{i=1}^{D} u(x_i, 10, 100, 4)\}$ $y_i = \quad 1 + \frac{x_i+1}{4}$ $k(x_i - a)^m \quad x_i > a$ $u(x_i, a, k, m) = \{\,0 \quad -a < x_i < a$ $k(-x_i - a)^m \; x_i < -a$	30	[-50,+50]	2000
$F_{13} = 0.1\{10\sin^2(\pi y_i)$ $+\sum_{i=1}^{D-1}(y_i - 1)^2\,[1 + 10\sin^2(\pi y_i + 1)]$ $+(yD - 1)^2 + \sum_{i=1}^{D} u(x_i, 10, 100, 4)\}$	30	[-50,+50]	2000
$F_{14} = [\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i - a_{ij})^6}]^{-1}$	2	[-65.53,+65.53]	150
$F_{15} = \sum_{i=1}^{11}[a_i - \frac{x_1(b_i^2+b_i x_i)}{b_i^2+b_1 x_3+x_4}]^2$	4	[-5,+5]	400
$F_{16} = 4x_1^2 - 2.1x_i^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	[-5,+5]	200
$F_{17} = (x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10(1 - \frac{1}{8\pi})\cos(x_1) + 10$	2	$[5,+10] \times [0,+15]$	180
$F_{18} = [1 + (x_1 + x_2 + 1)^2$ $(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)]$ $\times[30 + (2x_1 - 3x_2)^2$ $(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$	2	[-5,+5]	200
$F_{19} = -\sum_{i=1}^{4} c_i\, exp\,(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2)$	3	[0,+1]	100
$F_{20} = -\sum_{i=1}^{4} c_i\, exp\,(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2)$	6	[0,+1]	250
$F_{21} = -\sum_{i=1}^{5}[(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	[0,+10]	200
$F_{22} = -\sum_{i=1}^{7}[(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	[0,+10]	200
$F_{23} = -\sum_{i=1}^{10}[(X - a_i)(X - a_i)^T + c_i]^{-1}$	4	[0,+10]	200

Appendix B

The code sources used in this paper were downloaded from these addresses:

GWO: https://www.mathworks.com/matlabcentral/fileexchange/47258-grey-wolf-optimizer-toolbox

SCA: https://www.mathworks.com/matlabcentral/fileexchange/54948-sca–a-sine-cosine-algorithm
ABC and CA :http://yarpiz.com/

References

1. BoussaïD, I., J. Lepagnot, and P. Siarry, A survey on optimization metaheuristics. Information Sciences, 2013. 237: p. 82-117.

2. Mirjalili, S., S.M. Mirjalili, and A. Lewis, Grey wolf optimizer. Advances in Engineering Software, 2014. 69: p. 46-61.
3. Eberhart, R. and J. Kennedy. A new optimizer using particle swarm theory. in Micro Machine and Human Science, 1995. MHS'95., Proceedings of the Sixth International Symposium on. 1995. IEEE.
4. Karaboga, D., An idea based on honey bee swarm for numerical optimization, 2005, Technical report-tr06, Erciyes university, engineering faculty, computer engineering department.
5. Reynolds, R.G. An introduction to cultural algorithms. in Proceedings of the third annual conference on evolutionary programming. 1994. World Scientific.
6. Mirjalili, S., SCA: a sine cosine algorithm for solving optimization problems. Knowledge-Based Systems, 2016. 96: p. 120-133.

A hybrid meta-heuristic-based multi-agent for Industry 4.0

S. Tetouani[1,2], A. Chouar[1,2], A. Soulhi[2,3] and J. El-Alami[2,4]

[1.] *ESITH – CELog : Centre d'Excellence en LOGistique*
[2.] *LATIMI Laboratory:* Laboratoire d'Analyse des Systèmes, Traitement de l'Information et Management Intégré
Mohammed V University in Rabat-Morocco.
[3.] *High School of Mineral Industry, Rabat, Morocco*
[4.] *High School of Technology, Salé, Morocco*
E-mails: tetouani@esith.ac.ma; chaouar@esith.ac.ma; soulhi@gmail.com; elalami@gmail.com;

Abstract

E-commerce amplifies the direct linking from customers to manufacturers and enabling mass-personalization. So, the Industry 4.0 is able faster to make the personalized-product. In this paper, we present a hybrid meta-heuristic based multi-agent (called "hGATS+MAS") model for Industry 4.0. The objective is to schedule production orders as regards just-in-time with energy-efficiency. In this model, a Genetic Algorithm is made for a global search of the exploration space by a Master Scheduler Agent (MSA). Then, a local search is made by a Tabu Search to guide a set of Scheduler Agents (SAs).

Key words : Industry 4.0; Smart-Factory; Hybrid meta-heuristic; Multi-Agent System; Flexible Job Shop Scheduling Problem; Genetic algorithm; Tabu search

1. Introduction

The changes in the economic, political and technological evolution theses last years, have considerably affected our way to consume goods and services. The role of customers changed from a product-buyer to a product-designer himself [1]. In fact, E-commerce amplifies the direct linking from customers to manufacturers and enabling mass-customization or mass-personalization.

Industry 4.0 is able faster to offer the personalized-product. One major key of the Industry 4.0 is the "Smart-Factory" or the "Factory of the Future" (FoF) [2]. The Smart-Factory is a strongly automated and agile manufacturing system. Smart-Factory is piloted by an intelligent order fulfillment system [3], without human intervention (or almost) and manufactures for 24 hours a day and 7 days a week.

Mass-customization implies that the Smart-Factory is flexible, able to respond just-in-time delivery for a sustainable environment and with the energy-efficiency objectives. Theses multi-objectives are usually conflicting. In this case, how optimize scheduling *production orders* (PO) respecting simultaneously these different multi-objective constraints? An ideal solution which optimizes all the objective functions simultaneously cannot be found [4]. Multi-objective meta-heuristic algorithms are one of the most commonly used methods to obtain Pareto optimal solutions for a multi-objective optimization problem [5].

Furthermore, Industry 4.0 applies the multi-agent technology [6] that achieves complexity management, decentralization, intelligence, modularity, flexibility, robustness, adaptation, and responsiveness [7]. According to Nouri et al. (2018) [8] the multi-agent algorithm is an efficient and effective approach to solve the multi-objective problems, especially for the problems on a large scale. In this case, how multi-agent system (MAS) can be used to schedule *production orders* (PO)?

In other words, how combine multi-agent system (SMA) and multi-objective meta-heuristics to schedule *production orders* into Industry 4.0?

Consequently, to balance multi-objective scheduling constraints for Industry 4.0 we apply a hybrid meta-heuristics within a multi-agent system in two search levels. In a first level a genetic algorithm (GA) is developed to find global near-optimal initial-solutions. In the second level, a tabu search (TS) algorithm locates the best final solutions for each cluster. The hybrid genetic algorithm (GA) and the tabu search (TS) are implemented in two hierarchical multi-agent systems (hGATS+MAS).
The rest of this work is ordered as follows. In section 2, we provide a literature review on hybridization meta-heuristics into multi-agent system (MAS) to solve a flexible job shop scheduling problems. Section 3 presents briefly the formulation problem. Section 4 explains the hGATS+MAS proposed model. In section 5, we comment the experimental results. Finally, Section 6 provides the conclusions and the future works.

2. Literature review

Kacem et al., (2002) [9] proposed a Pareto approach based on the hybridization of fuzzy logic (FL) and evolutionary algorithms (EAs) to solve a flexible job-shop scheduling problem (FJSP). They minimize three objectives: makespan, total workload machines and maximum workload machine.

Wu and Weng (2005) [10] considered a multi-agent scheduling for a flexible job shop. They consider earliness and tardiness objectives. They develop two heuristic algorithms for jobs sequencing to deal with two kinds of jobs. The two kinds of jobs are defined to distinguish jobs with one operation left from jobs with more than one operation left. Different criteria are proposed to route these two kinds of jobs. The computational experiments show that the proposed multi-agent scheduling method significantly outperforms the existing scheduling methods in the literature. The proposed method is quite fast for a large scale.

Xia and Wu (2005) [11] applied the combination of s swarm optimization (PSO) and simulated annealing algorithm (SA) to solve the problem.

Gao et al. (2008) [12] have developed a hybridizing genetic algorithm with variable neighborhood descent. They use the ability of genetic algorithms (GA) for a global search. Then, they use the local search of variable neighborhood descent (VND). The hybrid-algorithm minimizes three objectives for a flexible job shop scheduling problems (FJSP): makespan, maximal workload machine and total workload machine.

Ennigrou and Ghédira (2005) [13] proposed two Multi-Agent approaches based on the Tabu Search (TS) meta-heuristic. They have distinguished between the global optimization approach where the tabu search (TS) has a macro vision on the system and the local optimization approach (FJS MATSLO) where the optimization distribute a collection of agents, each of them has its own local view.

Zhang et al. (2009) [14] proposed hybridizing the two optimization algorithms, particle swarm optimization (PSO) and tabu search (TS) algorithms, an effective hybrid approach for the multi-objective FJSP. PSO which integrates local search and global search scheme possesses high search efficiency. And TS which is a meta-heuristic designed to find a near optimal solution of combinatorial optimization problems.

Rajabinasab and Mansour (2010) [15] have studies a several dynamic events: stochastic processing times, uncertainty job arrivals, unpredicted machine breakdowns for a flexible job shop problem considering. A multi-agent scheduling system is a based on a pheromone approach. The simulation experiments are performed under various configurations: due date, utilization level, mean time to repair maintenance, breakdown level.

Moslehi and Mahnam (2011) [16] solve a multi-objective flexible job-shop scheduling problem by hybridization the birds' flight inspired particle swarm (Bio-PSO) and local search (LS) algorithm.

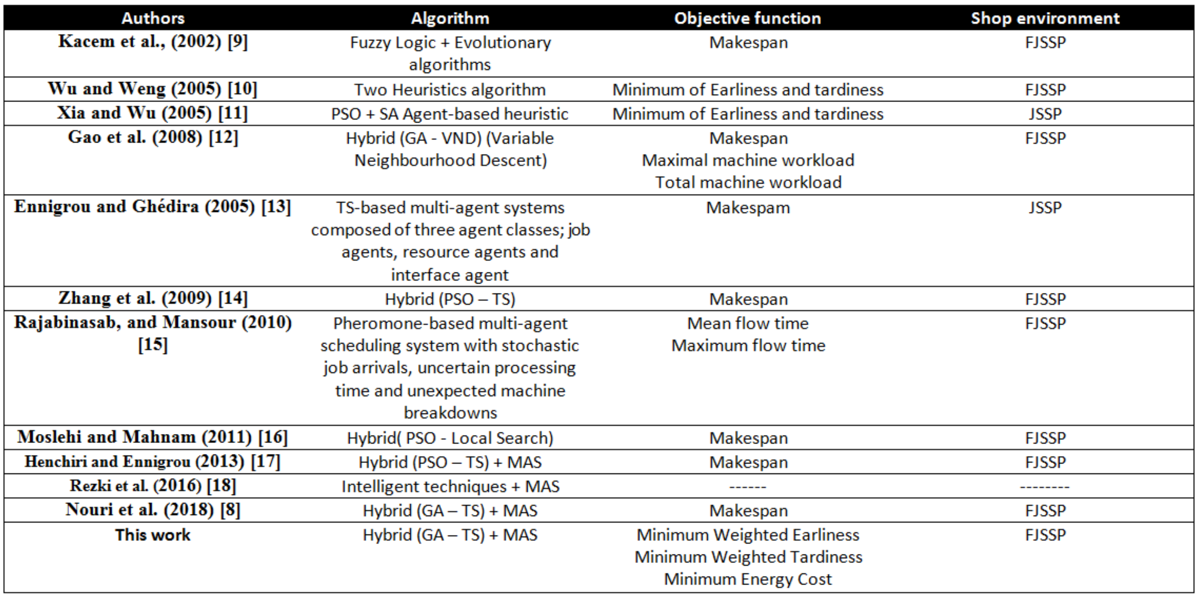
Henchiri and Ennigrou (2013) [17] developed a multi-agent model based on a hybridization of two meta-heuristics. A particle-swarm optimization (PSO) optimizes globally. The tabu search (TS) optimizes locally and gets a good exploitation of the good areas.

Rezki et al. (2016) [18] proposed a multi-agent system for complex process monitoring activities as detection, diagnosis, reconfiguration, and identification. They combine many intelligent techniques such as: multivariate control charts, neural networks, bayesian networks, and expert systems.

Nouri et al. (2018) [8] propose a hybrid metaheuristics-based clustered holonic multi-agent model to solve FJSP in two stages. A neighborhood-based genetic algorithm (NGA) is applied in the first stage. In this stage a scheduler agent explore a global search space. In the second stage, cluster-agents guide local searches using a tabu search algorithm. This hybridization technique improves the quality of the proposed model by decreasing computational time.

Table 1 summarizes the previous literature review. Our work is based on model of Nouri et al. (2018) [8] . However, our work differs from that of Nouri et al. (2018) [8] and other authors cited in the literature review, in that it integrates JIT and Energy consumption cost objectives to optimize simultaneously. In the next section, we formulate the problem.

Table 1: summary of the literature review



Authors	Algorithm	Objective function	Shop environment
Kacem et al., (2002) [9]	Fuzzy Logic + Evolutionary algorithms	Makespan	FJSSP
Wu and Weng (2005) [10]	Two Heuristics algorithm	Minimum of Earliness and tardiness	FJSSP
Xia and Wu (2005) [11]	PSO + SA Agent-based heuristic	Minimum of Earliness and tardiness	JSSP
Gao et al. (2008) [12]	Hybrid (GA - VND) (Variable Neighbourhood Descent)	Makespan Maximal machine workload Total machine workload	FJSSP
Ennigrou and Ghédira (2005) [13]	TS-based multi-agent systems composed of three agent classes; job agents, resource agents and interface agent	Makespam	JSSP
Zhang et al. (2009) [14]	Hybrid (PSO – TS)	Makespan	FJSSP
Rajabinasab, and Mansour (2010) [15]	Pheromone-based multi-agent scheduling system with stochastic job arrivals, uncertain processing time and unexpected machine breakdowns	Mean flow time Maximum flow time	FJSSP
Moslehi and Mahnam (2011) [16]	Hybrid(PSO - Local Search)	Makespan	FJSSP
Henchiri and Ennigrou (2013) [17]	Hybrid (PSO – TS) + MAS	Makespan	FJSSP
Rezki et al. (2016) [18]	Intelligent techniques + MAS	------	--------
Nouri et al. (2018) [8]	Hybrid (GA – TS) + MAS	Makespan	FJSSP
This work	Hybrid (GA – TS) + MAS	Minimum Weighted Earliness Minimum Weighted Tardiness Minimum Energy Cost	FJSSP

3. Problem Formulation

The scheduling production orders for a Smart-Factory in the context of the Industry 4.0 can be assimilated as a scheduling problem with three sub-problems. The first sub-problem is a just-in-time (JIT) scheduling into a Partial Flexible Job Shop Scheduling Problem (JIT-PFJS/SP). The second is Materials Handling (as AGVs, robots) Scheduling Problem (MH/SP) and finally the last, an Energy-Efficiency Scheduling Problem (EE/SP). However, in this work we do not take into account the transportation robots and we equate (very realistically) our problem as a just-in-time flexible job shop with energy-efficiency (JIT-FJS-EE/SP). In the following, we briefly describe the three sub-problems.

a. Flexible Job Shop Scheduling Problem (JIT-FJS/SP)

The flexible job shop scheduling problem (FJS-SP) could be formulated as follows. There is a set of *n* jobs J = $\{J_1, . . . , J_n\}$ to be processed on a set of m machines M = $\{M_1, . . . , M_m\}$. Each job J_i forms a sequence of n_i operations $\{O_{i,1}, O_{i,2}, . . . , O_{i,ni}\}$ to be executed successively according to the specified order. Each alternative machines $M(O_{i,j})$ is capable of processing each operation $O_{i,j}$.

The ideal JIT schedule is one in which all jobs finish exactly on their assigned due dates. Job that completes early is storage inventory until its due date, whereas a job that completes after its due date can lead to the loss of a customer. For that reason, the Earliness and Tardiness represent excellent keys performance indicators [19], [20].

We think a JIT scheduling and for each operation O_{ij} we have two penalty factors: α_{ij} and β_{ij}. Early operation is penalized by the cost $\beta_{ij}E_{ij}$ and tardy operation by $\alpha_{ij}T_{ij}$. The purpose is to minimize, for all operations, the total costs of earliness and tardiness simultaneously.

$$\min f_1 = \sum_{i=1}^{n}\sum_{j=1}^{n_i}\left(\beta_{ij}. \max(0, d_{ij} - C_{ij})\right) \text{ (1)}$$

$$\min f_2 = \sum_{i=1}^{n}\sum_{j=1}^{n_i}\left(\alpha_{ij}. \max(0, d_{ij} - C_{ij})\right) \text{ (2)}$$

α_{ij}: unit tardiness penalty cost for job j
β_{ij}: unit earliness penalty cost for job j
C_{ij} : *Completion time of job j*
d_{ij}*: due-date of job j*

b. Energy-Efficiency Scheduling Problem (EE/SP)

A Time-of-use electricity (TOU) tariff is a good opportunity for electricity consumers to reduce their energy bills by shifting their usage from on-peak periods to off-peak ones [21]. Time-of-use strategy offers variable electricity tariffs over time: high price in on-peak periods and low price in off-peak periods [22]. Scheduling the production orders in a planning horizon, according TOU tariffs decrease the total electricity cost strongly.

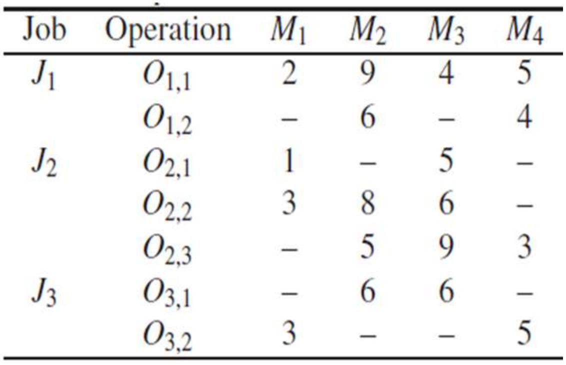
The calculation of total energy consumption (TEC) is indicated by Equation (3) and it considers two machine modes: processing, and idle-sleep mode.

$$\min f_3 = \sum_{t=1}^{r}\sum_{k=1}^{m} EC_t \left(\sum_{s=1}^{q} E_{stk} + E_{otk}\right)\text{(3)}$$

E_{stk} *processing energy consumption on machine at speed s at period t*
E_{otk} *idle-sleep energy consumption on machine k at period t*

To make clear the JIT-FJS-EE/SP, table 2 shows a sample problem of three jobs and four machines. The number corresponds to a processing-time and "-"indicate that the operation cannot be processed on the related machine. Figure 1 shows an example of an optimal scheduling with regard to the earliness, tardiness and energy efficiency constraints.

Table 2 : Example of instance for a FJSP



Job	Operation	M_1	M_2	M_3	M_4
J_1	$O_{1,1}$	2	9	4	5
	$O_{1,2}$	–	6	–	4
J_2	$O_{2,1}$	1	–	5	–
	$O_{2,2}$	3	8	6	–
	$O_{2,3}$	–	5	9	3
J_3	$O_{3,1}$	–	6	6	–
	$O_{3,2}$	3	–	–	5

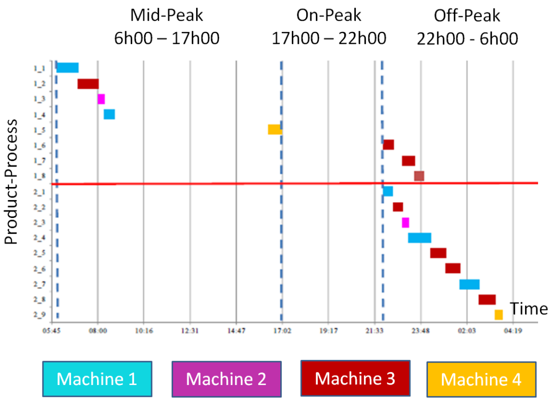


Figure 1: Example of optimal scheduling

4. Hybrid meta-heuristic-based multi-agent system

A multi-agent system (MAS) consists a collection of individual agents, each agent displays a certain amount of autonomy with respect to its actions and perception of a domain. This work is based on model of Nouri et al. (2018) [8] and we use a hybrid meta-heuristic approach processing in two levels: global search level with a genetic algorithm and (GA) generating initial–solutions and local search level with tabu search (TS) starting from the initial-solutions.

At the first level, a global exploration use a genetic algorithm (GA) to find feasible zones in the search space and a aggregating operator allowing regrouping them in a "set of clusters". The choice of GA is motivated by the fact that a GA is able to produce several different faster near optimal solutions [23]. In fact, in a GA, each candidate solution to a problem (know as a "chromosome") is an ordered fixed-length array of values for attributes ("genes").

In the second level, a Tabu Search (TS) algorithm is implemented to near-locate the best individual solution for each cluster. The choice of Tabu Search (TS) is justified by the fact it produces better solution, with less computing time, than other meta-heuristics [24]. Indeed, Tabu Search (TS) algorithm is a meta-heuristic method search starting from an initial solution as current solution and using internal memory [25]. The advantage of TS over other methods is to use its internal memory to prevent from searching previous visited areas. Therefore, it is easier to lead towards local optimum in sort time.

The operation method of the proposed model is implemented in two hierarchical multi-agent systems, named hybrid Genetic Algorithm with Tabu Search within a Multi-agent System (hGATS+MAS), see Figure 2.

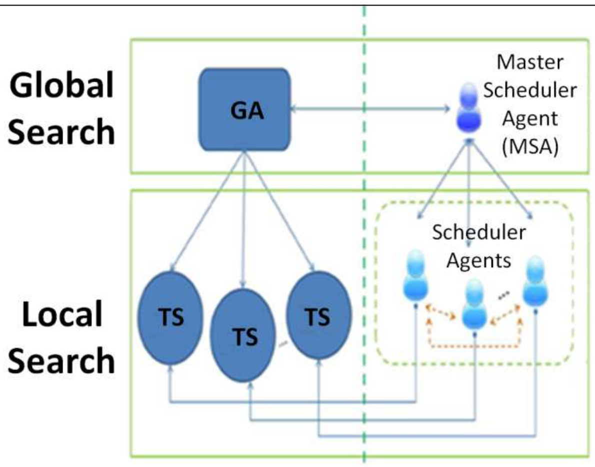


Figure 2: Hybrid meta-heuristic based Multi-Agent System (hGATS+MAS) [8]

The first multi-agent level is composed by a Master Scheduler Agent (MSA). This mission is to prepare globally the near-optimal clusters (or regions) of the search space. The second multi-agent level contains a set of Scheduler Agents (SAs). There missions is to provide the best local solution into the global near-optimal solution.

A. Master Scheduler Agent

The role of the Master-Scheduler Agent (MSA) is to identify regions with high average fitness. The search space is done during a fixed number of iterations. Then, a clustering operator is integrated to divide the best identified areas by the GA in the search space to different parts where each part is a cluster. Then, MSA creates *N* Scheduler Agents (SAs) to organize the passage to the next step of the global algorithm. After that, the MSA remains in a "stop state" until to receipt the optimal solutions provide from the SA for each cluster. The process it finally finishes by providing the final solution.

B. Scheduler Agent

To each cluster, each Scheduler Agent (SAs) applies individually a local search technique by a Tabu Search algorithm. The tabu search conduct the research into their search space improves the quality from the final population of the genetic algorithm. In fact, this process is executed simultaneously by all the set of the SAs agents. Each SA starts the research from a near-optimal solution of its cluster searching to arrive at new most important individual solutions separately in its assigned cluster.

5. Experimental Results

The proposed (hGATS+MAS) is implemented in java language. The tests are conducted on PC with an Intel^{R} Core^{TM} i5-4300U 1.9-GHz CPU with 8 GB of RAM memory. We use the Integrated Development Environment (IDE) *eclipse* We code the algorithm and the multi-agent into a platform *Jade*22.

To test the proposed model, we simulate a scheduling instance for 6 machines and 8 jobs. We apply here 4 situations with 30 replications. To stress the algorithm, we used the following constraints: the availability date of all the POs is fixed at 22:00 at the latest.

Figure 3a shows the configurations where Tardiness penalty cost is the most important, the rest are all equal. In this case the completion time is minimized. Both Figure 3b and 3c, shows configurations where Earliness penalty cost is most important. In Figure 3d we show planning scheduling where energy cost is increasing, so the jobs are sequencing in the way to circumvent on-peak electricity cost.

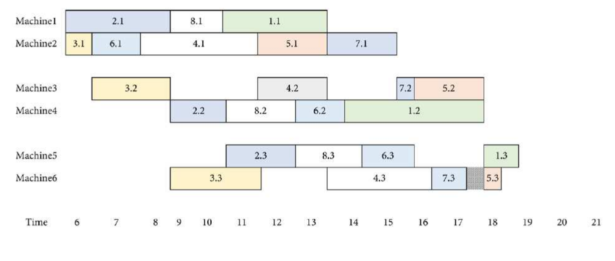


Figure 3a: High Tardiness penalty cost

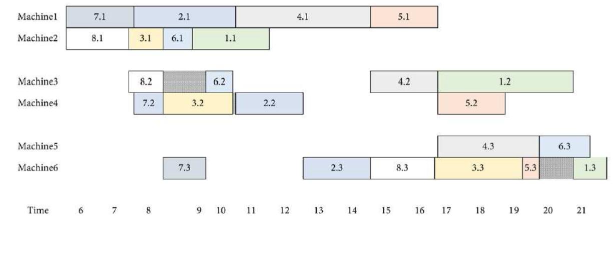


Figure 3b: High Earliness penalty cost and low Energy cost

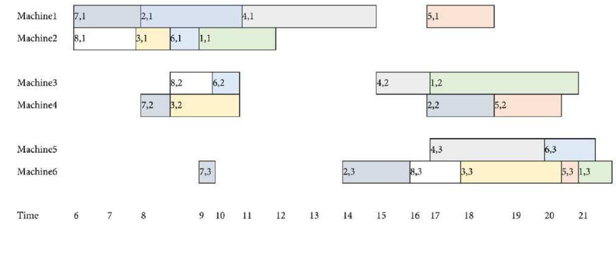


Figure 3c: High Earliness penalty cost and High Energy cost

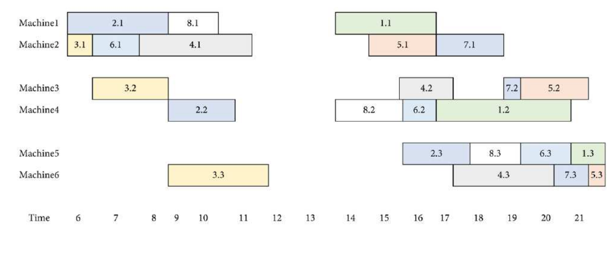


Figure 3d: High Energy Cost

6. Conclusion

In this paper, we present a hybrid meta-heuristic based multi-agent (hGATS+MAS) model for Industry 4.0. The objective is to schedule production orders as regards just-in-time with energy-efficiency. In this model, a Genetic Algorithm is made for a global search of the exploration space by a Master Scheduler Agent (MSA). Then, a local search is made by a Tabu Search to guide a set of Scheduler Agents (SAs).

We tested our model with hard constraints and it responds in a very low time. However, we did not perform a benchmark test to compare its effectiveness. In the very near future, we look forward to comparing our model with instances from the literature. Finally, we will search to treat other configurations of the FJSP, such as by integrating materials handling (robots, AVGs) constraints in the shop floor, maintenance scheduling and other extensions of energy-efficiency as power consumption.

References

[1] S. Kaplan et M. Sawhney, « E-hubs: the new B2B marketplaces », *Harv. Bus. Rev.*, vol. 78, n^o 3, p. 97‑106, 2000.

[2] M. Hermann, T. Pentek, et B. Otto, « Design principles for industrie 4.0 scenarios », in *System Sciences (HICSS), 2016 49th Hawaii International Conference on*, 2016, p. 3928‑3937.

[3] R. Harrison, D. Vera, et B. Ahmad, « Engineering the smart factory », *Chin. J. Mech. Eng.*, vol. 29, n^o 6, p. 1046‑1051, 2016.

[4] A. Zhou, B.-Y. Qu, H. Li, S.-Z. Zhao, P. N. Suganthan, et Q. Zhang, « Multiobjective evolutionary algorithms: A survey of the state of the art », *Swarm Evol. Comput.*, vol. 1, n^o 1, p. 32‑49, 2011.

[5] K. Genova, L. Kirilov, et V. Guliashki, « A survey of solving approaches for multiple objective flexible job shop scheduling problems », *Cybern. Inf. Technol.*, vol. 15, n^o 2, p. 3‑22, 2015.

[6] A. Rocha *et al.*, « An agent based framework to support plug and produce », in *Industrial Informatics (INDIN), 2014 12th IEEE International Conference on*, 2014, p. 504‑510.

[7] T. Nagata et H. Sasaki, « A multi-agent approach to power system restoration », *IEEE Trans. Power Syst.*, vol. 17, n° 2, p. 457‑462, 2002.

[8] H. E. Nouri, O. B. Driss, et K. Ghédira, « Solving the flexible job shop problem by hybrid metaheuristics-based multiagent model », *J. Ind. Eng. Int.*, vol. 14, n° 1, p. 1‑14, 2018.

[9] I. Kacem, S. Hammadi, et P. Borne, « Pareto-optimality approach for flexible job-shop scheduling problems: hybridization of evolutionary algorithms and fuzzy logic », *Math. Comput. Simul.*, vol. 60, n° 3‑5, p. 245‑276, 2002.

[10] Z. Wu et M. X. Weng, « Multiagent scheduling method with earliness and tardiness objectives in flexible job shops », *IEEE Trans. Syst. Man Cybern. Part B Cybern.*, vol. 35, n° 2, p. 293‑301, 2005.

[11] W. Xia et Z. Wu, « An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problems », *Comput. Ind. Eng.*, vol. 48, n° 2, p. 409‑425, 2005.

[12] J. Gao, L. Sun, et M. Gen, « A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems », *Comput. Oper. Res.*, vol. 35, n° 9, p. 2892‑2907, 2008.

[13] M. Ennigrou et K. Ghédira, « New local diversification techniques for flexible job shop scheduling problem with a multi-agent approach », *Auton. Agents Multi-Agent Syst.*, vol. 17, n° 2, p. 270‑287, 2008.

[14] G. Zhang, X. Shao, P. Li, et L. Gao, « An effective hybrid particle swarm optimization algorithm for multi-objective flexible job-shop scheduling problem », *Comput. Ind. Eng.*, vol. 56, n° 4, p. 1309‑1318, 2009.

[15] A. Rajabinasab et S. Mansour, « Dynamic flexible job shop scheduling with alternative process plans: an agent-based approach », *Int. J. Adv. Manuf. Technol.*, vol. 54, n° 9‑12, p. 1091‑1107, 2011.

[16] G. Moslehi et M. Mahnam, « A Pareto approach to multi-objective flexible job-shop scheduling problem using particle swarm optimization and local search », *Int. J. Prod. Econ.*, vol. 129, n° 1, p. 14‑22, 2011.

[17] A. Henchiri et M. Ennigrou, « Particle swarm optimization combined with tabu search in a multi-agent model for flexible job shop problem », in *International Conference in Swarm Intelligence*, 2013, p. 385‑394.

[18] N. Rezki, O. Kazar, L. H. Mouss, L. Kahloul, et D. Rezki, « On the use of multi-agent systems for the monitoring of industrial systems », *J. Ind. Eng. Int.*, vol. 12, n° 1, p. 111‑118, 2016.

[19] R. Rios et Y. A. Ríos-Solís, *Just-in-time Systems*, vol. 60. Springer Science & Business Media, 2011.

[20] J. L. García-Alcaraz et A. A. Maldonado-Macías, *Just-in-time elements and benefits*. Springer, 2016.

[21] Y. Wang et L. Li, « Time-of-use based electricity demand response for sustainable manufacturing systems », *Energy*, vol. 63, p. 233‑244, 2013.

[22] J.-Y. Moon, K. Shin, et J. Park, « Optimization of production scheduling with time-dependent and machine-dependent electricity cost for industrial energy efficiency », *Int. J. Adv. Manuf. Technol.*, vol. 68, n° 1‑4, p. 523‑535, 2013.

[23] K. Deb, A. Pratap, S. Agarwal, et T. Meyarivan, « A fast and elitist multiobjective genetic algorithm: NSGA-II », *IEEE Trans. Evol. Comput.*, vol. 6, n° 2, p. 182‑197, 2002.

[24] G. Vilcot et J.-C. Billaut, « A tabu search and a genetic algorithm for solving a bicriteria general job shop scheduling problem », *Eur. J. Oper. Res.*, vol. 190, n° 2, p. 398‑411, 2008.

[25] F. Pezzella, G. Morganti, et G. Ciaschetti, « A genetic algorithm for the flexible job-shop scheduling problem », *Comput. Oper. Res.*, vol. 35, n° 10, p. 3202‑3212, 2008.

A constructive heuristic for two machine job shop scheduling problem under availability constraints on one machine

M. Benttaleb[1], F. Hnaien[1], and F. Yalaoui[1]

ICD-LOSI, Université de Technologie de Troyes, UMR CNRS 6281, 12 Rue Marie Curie, CS 42060-10004
Troyes Cedex - France
mourad.benttaleb@utt.fr faicel.hnaien.fr farouk.yalaoui@utt.fr

1 Introduction and related works

We present in this paper an approximation approach for solving the two-machine job shop scheduling problem under availability constraints one one machine. In real industrial settings, it is often necessary to address situations where the machines become unavailable during periods of time throughout the scheduling horizon for different reasons. In this context, machines may be unavailable due to deterministic events such as preventive maintenance.

For the studied problem, a set of jobs and two machines are considered. Each job has to be processed given an order through the two machines. The processing times of all operations are known. Every job is to be processed on at most one machine at a time and each machine performs no more than one job at a time. Machines are not continuously available for processing. Further, the beginnings and durations of each unavailability period is known in advance and operations are not able to be preempted neither by unavailability periods nor by other operations. The goal is to minimize the makespan, i.e., the maximum completion time of all jobs on the two machines.

In many industrial application such as Additive and Subtractive manufacturing implying the use of machine tools or machine shops, the manufacturing process could be modeled on two machine jobs shop scheduling problem. The application could concern also conveyor transport, robotic transportation, Material handling or conditioning and many other problems.

Literature considers various cases relating to the processing of an operation interrupted by an unavailability period. Lee [1] introduces three types of operations. An operation that cannot be finished before the next unavailability period of a machine is defined as *resumable* if it can be continued after the machine is available again and *non-resumable* where it must be totally restarted. The *semi-resumable* operation must be partially restarted when th machine become available. [2] defines the problem when the processing of an operation cannot be interrupted neither by the starting of another operation, nor by an availability period as strictly as *non-preemptive* job shop scheduling problem with availability constraints. [3] defines unavailability period that allow an operation to be interrupted as *crossable* period and the one that prevent interruption as *non-crossable* period.

It is well known that the two-machine job shop problem with no availability constraints is solvable optimally and polynomially by Jackson's algorithm [4]. However, it becomes strongly NP-Hard if the machines are not continuously available. Lee [5, 1] proves that the *resumable*, *non-resumable* or *semi-resumable* two-machine flow shop problem with a one availability constraint on one machine is NP-hard for each of these cases.

In [6], we present an heuristic which is the application of the Jackson's algorithm when the scheduling of jobs take into account the unavailability periods that their duration and starting time are known in advance. we have also demonstrated that the worst case ratio is equal to 2. The retained results deduce that the more the number of unavailabilities increases the more the quality of the solution obtained approximately deteriorates considering the number of idles times generated which is equal to the number of unavailability period. In [6], it is proved that there exists an optimal order guaranteeing that jobs before, after and between each two consecutive unavailability periods are ordered following Jackson's rule, for two-machine job shop with unavailability periods on one machine.

Facing the requirement to obtain good solutions for problems, we develop a constructive heuristic method to deal with the two machine job shop under availability constraints and we compare computational results with the heuristic $H1$ provided in [6].

2 Approach

The construction heuristic (H_{constr}) uses Jackson's rule in its approach. First, the order of jobs is found according to this rule (JKO). The choice of the unavailable machine is completely similar as long as the two-machine scheduling problem is symmetrical. We consider that machine M_1 is the unavailable for periods.

Each job in M_1 is affected into a suitable interval following JKO order. The first job candidate is inserted in the first interval, If the completion time of the affected job on machine M_1 is greater than the starting time of the unavailability, the job is assigned to the following interval, until its final assignment. The procedure repeats for te next job following JKO order. The algorithm ends when all jobs are affected. When the set of jobs to be executed in each interval is defined, we use the order proved in [6] (Property 6) that schedules these sets and guaranteeing that of jobs order in each interval is optimal.

3 Computational results and discussion

The proposed algorithms are coded in the $C++$ language. We use for run a PC with 2.60GHz Intel(R) Core (TM) i5-4210M CPU and 8.00 GB, on Linux operating system.

We employed the (15 jobs/15 machines, 20 jobs/15 machines, 30 jobs/15 machines) Taillard's job shop instances [7]. We took the first two values of each job, corresponding to its durations on the two first machines for each instance.

Further, we add the parameters concerning period of unavailability to be fixed on machine M_1. The duration of the u^{th} unavailability period is assumed equal to the average of the processing times of the operations on machine $M_1 : g_u = t_u - s_u = \frac{\sum_{i=1}^{N} p_{i1}}{N}$.

We generate instances considering the location of the u^{th} unavailability period. The starting date of this period is set according to the total number of unavailability periods considered on machine $M_1 : s_u = \frac{u}{U+1} \sum_{i \in N} p_{i1} - \frac{1}{2} g_u$.

Table 1 illustrates the heuristic performance (H_{constr}) proposed in this paper with respect to the $H1$ heuristic. It presents makespan (C_{max}) the gap to lower bound ($Gap = \frac{C_{max}-LB}{LB}(\%)$) and the execution time by each heuristic ($CPU(s)$). Figure 1 summarizing these results, indicates clearly that the heuristic proposed in this paper outperforms $H1$ proposed in [6]. Obviously, these good performance of the proposed approach is due to the fact that H_{constr} consider in each job's assignment all intervals $(I_1, ..., I_{U+1})$ so each idle time that can not be filled by a defined job could be used by one of the next jobs to assign. Furthermore, the average gap displayed by the figure is determined related to the lower bound (see [6]) which is not necessarily the optimal solution.

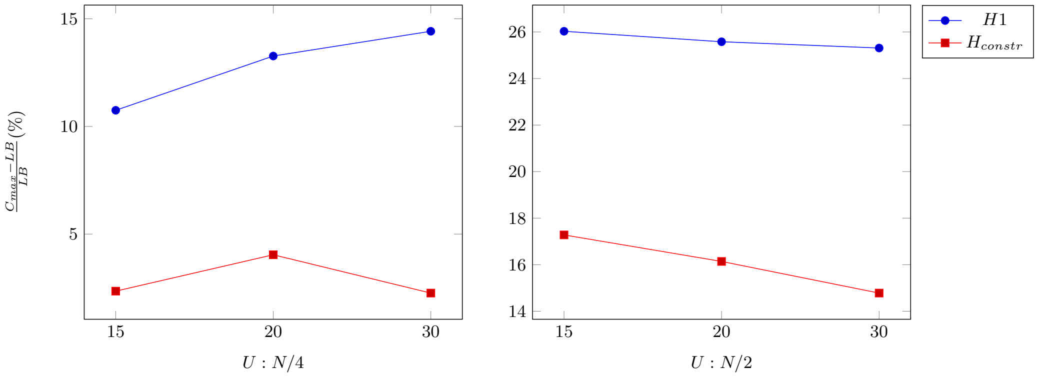


Fig. 1. Average gap $\frac{C_{max}-LB}{LB}$ (%) for each instance size ($N = \{15, 20, 30\}$, $U\{N/4, N/2\}$)

Table 1. Comparison results, H_{constr} and $H1$ (Taillard instances $N = \{15, 20, 30\}$, $P3$)

N	$U = N/4$							$U = N/2$						
	LB	$H1$			H_{constr}			LB	$H1$			H_{constr}		
		C_{max}	$Gap(\%)$	$CPU(s)$	C_{max}	$Gap(\%)$	$CPU(s)$		C_{max}	$Gap(\%)$	$CPU(s)$	C_{max}	$Gap(\%)$	$CPU(s)$
15	902	902	0.00	<0.001	902	0.00	0.002	938	1163	23.99	<0.001	1083	15.46	0.001
	819	939	14.65	<0.001	824	0.61	0.001	999	1277	27.83	<0.001	1095	9.61	0.002
	936	1005	7.37	<0.001	949	1.39	0.001	1144	1463	27.88	<0.001	1348	17.83	0.002
	852	895	5.05	<0.001	855	0.35	0.001	1003	1250	24.63	<0.001	1139	13.56	0.002
	972	1082	11.32	<0.001	988	1.65	0.001	1188	1499	26.18	<0.001	1383	16.41	0.002
	983	1064	8.24	<0.001	1024	4.17	0.001	1199	1488	24.10	<0.001	1424	18.77	0.003
	1084	1263	16.51	<0.001	1102	1.66	0.001	1324	1643	24.09	<0.001	1564	18.13	0.002
	967	1182	22.23	<0.001	993	2.69	0.001	1179	1497	26.97	<0.001	1447	22.73	0.002
	1109	1235	11.23	<0.001	1205	8.66	0.002	1353	1740	28.60	<0.001	1665	23.06	0.002
	800	911	13.88	<0.001	825	3.13	0.001	976	1195	22.44	<0.001	1125	15.27	0.001
20	1268	1458	14.98	<0.001	1358	7.10	0.002	1518	1862	22.66	<0.001	1739	14.56	0.004
	1178	1334	13.24	<0.001	1199	1.78	0.002	1413	1734	22.72	<0.001	1631	15.43	0.004
	1265	1425	12.65	<0.001	1347	6.48	0.002	1515	1921	26.80	<0.001	1777	17.29	0.003
	1293	1433	10.83	<0.001	1319	2.01	0.002	1548	1940	25.32	<0.001	1745	12.73	0.003
	1082	1215	12.29	<0.001	1108	2.40	0.002	1297	1656	27.68	<0.001	1505	16.04	0.003
	1332	1485	11.49	<0.001	1340	0.60	0.002	1597	2020	26.49	<0.001	1875	17.41	0.004
	1387	1595	15.00	<0.001	1452	4.69	0.002	1662	2117	27.38	<0.001	1901	14.38	0.003
	1326	1534	15.69	<0.001	1397	5.35	0.002	1591	1970	23.82	<0.001	1890	18.79	0.004
	1291	1462	13.25	<0.001	1368	5.96	0.002	1546	1969	27.36	<0.001	1834	18.63	0.004
	1433	1650	15.14	<0.001	1494	4.26	0.002	1718	2164	25.96	<0.001	2075	20.78	0.004
30	2053	2367	15.29	<0.001	2119	3.21	0.004	2493	3167	27.04	<0.001	2819	13.08	0.008
	1964	2355	19.91	<0.001	1986	1.12	0.004	2388	3050	27.72	<0.001	2799	17.21	0.009
	1945	2164	11.26	<0.001	1993	2.47	0.003	2361	2937	24.40	<0.001	2840	20.29	0.009
	1828	2234	22.21	<0.001	1849	1.15	0.004	2220	2817	26.89	<0.001	2586	16.49	0.007
	1967	2152	9.41	<0.001	2031	3.25	0.004	2391	2975	24.42	<0.001	2690	12.51	0.007
	1975	2211	11.95	<0.001	2015	2.03	0.004	2399	3060	27.55	<0.001	2759	15.01	0.009
	2184	2454	12.36	<0.001	2249	2.98	0.005	2656	3307	24.51	<0.001	2962	11.52	0.009
	1768	2020	14.25	<0.001	1815	2.66	0.003	2144	2622	22.29	<0.001	2420	12.87	0.008
	1784	2018	13.12	<0.001	1810	1.46	0.004	2168	2667	23.02	<0.001	2472	14.02	0.007
	1890	2183	15.50	<0.001	1944	2.86	0.004	2298	2932	27.59	<0.001	2754	19.84	0.008

4 Conclusion

We present a constructive heuristic based on Jackson's rule, to resolve approximately the non-preemptive two-machine job shop scheduling problem under availability constraints on one machine. The proposed heuristic outperforms the one presented previously in [6]. This will allow to obtain a quality approximate solution in a very small computational time (ms). Obviously, it helps to get a good upper bound for the exact resolution of the problem.

References

1. Lee C-Y. Two-machine flowshop scheduling with availability constraints. European Journal of Operational Research 1999; 114: 420-429.
2. Aggoune, R., 2002. Ordonnancement d'ateliers sous contraintes de disponibilité des machines. Ph.D. thesis, Université de Metz.
3. Mauguière, P., Billaut, J.-C., Bouquard, J.-L., 2005. New single machine and job-shop scheduling problems with availability constraints. Journal of scheduling 8 (3), 211-231.
4. Jackson, J. R., 1956. An extension of johnson's results on job idt scheduling. Naval Research Logistics Quarterly 3 (3), 201-203.
5. Lee C-Y. Minimizing the makespan in the two-machine flowshop scheduling problem with an availability constraint. Operations Research Letters 1997; 20:129-39
6. Benttaleb, Mourad, Faicel Hnaien, and Farouk Yalaoui. 2018. Two-machine job shop problem under availability constraints on one machine: Makespan minimization. Computers & Industrial Engineering 117: 138 - 151.
7. Taillard, E., 1993. Benchmarks for basic scheduling problems. European journal of operational research 64 (2), 278-285.

A heuristic rule for ranking scientific journals based on citation impact

T. Lando[1] and L.Bertoli-Barsotti[2]

1. VŠB-Technical University of Ostrava, Sokolskà Trida 33, Ostrava (Czech Republic); University of Bergamo, via dei Caniana 2, Bergamo (Italy). tommaso.lando @unibg.it

2. University of Bergamo, via dei Caniana 2, Bergamo (Italy). lucio.bertoli-barsotti @unibg.it

Keywords : citation impact, ranking.

Abstract

Recently, the scientific community has given a lot of attention to research evaluation. In this framework, one of the major issues in the scientometrics literature is ranking scientific journals by quantifying their "impact", based on citation data. Many methods have been introduced to this purpose, among which we can find, for instance, the well-known "impact factor", which is basically an average of citations. However, a rigorous approach to this ranking problem consists of performing a functional comparison between citation distributions, unlike a scalar comparison of indices. This gives rise to a pre-order relation, i.e. the second-degree stochastic dominance, also denoted as generalized Lorenz dominance (GLD) [1], which makes it possible to rank a large part of the distributions. Unfortunately, pre-orders are not complete orders, so that, within a given set, there may be many pairs of distributions that cannot be ranked by GLD. Moreover, the verification of the GLD may represent an issue from a computational point of view. We show with a heuristic argument that the citation distributions of journals can be described by some basic statistical tools, such as the mean and the Gini index. In particular, we argue that knowledge of the mean and the Gini index may be sufficient to provide a good approximation of the verification of the dominance relations that may be (or may be not) satisfied among a set of pairs of journals. Therefore, we propose a rule that may be used to find the dominant journal, in terms of impact, within a given set, and, consequently, we introduce an algorithm to implement such rule. Empirical results show that the method provides a good approximation and can therefore yield a reliable ranking.

Funding

This paper has been supported through the Czech Science Foundation (GACR) under project 17-23411Y(to T.L.).

References

[1] Lando, T., & Bertoli-Barsotti, L. (2017). Measuring the citation impact of journals with generalized Lorenz curves. Journal of Informetrics, 11(3), 689-703.

Assessing Film Coefficients of Microchannel Heat Sinks Through the Cuckoo Search Algorithm

Jorge M. Cruz-Duarte[1], Arturo Garcia-Perez[1], Iván Amaya[2], Rodrigo Correa[3]

[1] *Universidad de Guanajuato, Division de Ingenierías del Campus Irapuato-Salamanca, Salamanca, Guanajuato, México.*
jorge.cruz@ugto.mx, arturo@ugto.mx

[2]*Escuela de Ingeniería y Ciencias, Tecnologico de Monterrey, Campus Monterrey, México.*
iamaya2@itesm.mx

[3] *Universidad Industrial de Santander, Escuela de Ingenierías Electrica, Electronica y de Telecomunicaciones,*
Bucaramanga, Colombia. crcorrea@uis.edu.co

Abstract

Film transfer coefficient is one of the most challenging variables to measure in experimental heat transfer. This happens because such variable depends on too many others. Examples include type of media (gas or liquid), body geometry, fluid flow, thermal conductivity, and many more thermodynamic properties. In this paper, we propose estimating such coefficient by solving an *inverse heat transfer problem* via the Cuckoo Search global optimization algorithm. The designs were achieved through the *entropy generation minimization criterion*, also powered by Cuckoo Search, employing several specifications (material, working fluid and $\dot{Q}$). Our data show great estimations for signal-to-noise ratios above 30 dB, which can be reached with virtually any modern temperature sensors.

Keywords: inverse heat transfer problem; rectangular microchannel heat sink; Cuckoo Search; global optimization.

Introduction

Thermal management of modern electronic devices is a well–known problem nowadays. Such problem arises during design and operation processes [1,2]. It is easy to find many reported solutions including theoretical, numerical and experimental studies [3-6]. Specifically, microchannel heat sinks (MCHSs) have been employed on this problem, and they are recurrently implemented in high thermal power dissipation applications [7-9]. This idea began with an early implementation proposed by Tuckerman and Pease in [10]. MCHSs mainly ensure a maximal dissipation of electronic power losses and a minimal additional energy consumption. For that reason, multiple MCHS design strategies have appeared in literature [11-13]. Among the most recent and powerful ones resides the one pioneered by Bejan [14]. His approach focuses on the thermodynamics-based Entropy Generation Minimization (EGM) criterion.

Every practical electronic thermal management system, such as a MCHS, may have its performance altered when operating in a noisy environment. Moreover, practical engineering applications require precise estimation of some parameters, parting from temperature measurements. Some examples include thermophysical properties of materials and coolants, as well as boundary and initial conditions, and energy source distributions. This kind of problem is known as an Inverse Heat Transfer Problem (IHTP) [15]. Its solution can help study the differences between theoretical and real system performance. For that reason, after implementing a MCHS, as well as any practical engineering system, it is necessary to add an estimation procedure for eventually tuning the system. This can help prevent marginal or unexpected behaviors.

IHTP have been broadly used on diverse areas of practical engineering, with very creative methodologies. For example, Wang *et al.* estimated the grinding heat flux from a 2D temporal and spatial temperature distribution, throughout a surgical bone grinding process, implementing sequential function specification method and sequential programming [16]. Li *et al.* determined multiple heat sources for a mold heating system of the injection machine by solving a 3D inverse heat conduction problem [17]. Cornejo *et al.* approached the thermophysical properties of food in the freezing temperature range, by implementing an inverse method based on simulation results [18]. Wang *et al.* determined heat transfer coefficients in a steel billets continuous casting process under large disturbance. They used an approach based on weighted least squares and Levenberg-Marquardt (LM) [19]. Mohebbi *et al.* accurately determined thermal conductivity as a linear temperature function in a 2D steady-state heat conduction process [20]. Luo and Yang calculated the total heat exchange factor by solving the IHTP via a combined method based on gradients. They employed it as reference trajectories for a control system in a reheating furnace [21]. On the other hand, Chanda *et al.* reached a practical thermal conductivity tensor of anisotropic composite materials, via a characterization methodology powered by an artificial neural network and a genetic algorithm [22]. Dhiman and Prasad found empirical correlations for local and average Nusselt numbers of a heated hollow cylinder in cross flow of air, implementing the LM method [23]. Furthermore, other works have been aimed at improving traditional IHTP solving methodologies, by including Tikhonov regularization. An alternative inversion model was studied by Mu *et al.* They solve an optimization problem by splitting it out in several simpler sub-problems, and then tackling each one through the Broyden-Fletcher-Goldfarb-Shanno algorithm [24]. Additionally, Duda proposed an inverse method to avoid the stability drawbacks of classical methods when solving IHTPs while using real-time measurements [25].

Nevertheless, very few manuscripts have reported IHTP solutions on heat sinks applications. Bodla *et al.* found the optimal ranges for uncertainties related to geometry and operating conditions in fin heat sinks using a stochastic methodology [26,27]. Huang *et al.* estimated the optimal perforation diameter of a perforated pin-fin heat sink for a desired temperature reference [28]. Sudheesh and Siva characterized some parameters associated to heat transfer in trailing heat sinks for the reduction of stresses and distortions from welded structures [29]. Chen *et al.* determined average convective heat transfer coefficients of an operating plate-fin heat sink, under different fluid flow conditions. They solved the IHTP with the help of FLUENT simulation data [30, 31]. More specific works have dealt with microchannel heat sinks, such as the one of Huang *et al.*, who accurately estimated the dynamic heat transfer coefficient in a multi-microchannel evaporator under disturbance. Authors approached the three-dimensional IHTP using a combined strategy of the tridiagonal matrix algorithm, Newton-Raphson, and local energy balance. Moreover, Maciejewska and Piasecka determined the thermal coefficients involved on a deep minichannel vertically oriented and filled with an electronic cooling liquid, using the Trefftz method.

As aforementioned, most works focus on the use of gradient-based techniques. Thus, we strive to fill the knowledge gap regarding the performance of modern optimization techniques for these kind of problems. Hence, we use an alternative methodology for solving inverse heat transfer problems. As stated, we consider a modern optimization algorithm instead of gradient-based techniques. To illustrate this methodology, average convective heat transfer coefficients were estimated for different scenarios. We analyze several noise levels by creating different synthetic temperature measurements. To solve the IHTP, we follow an approach that uses the least-square error (LSE) criterion as the objective function, and the Cuckoo Search (CS) algorithm as an optimizer. Noiseless temperature data were obtained from solving a forward problem. Such problem represents a MCHS designed under certain specifications with the EGM criterion, also powered by CS. Results show great estimations for signal-to-noise (SNR) values above 30 dB.

This paper starts by describing the heat transfer problems, i.e. the thermal management scenario followed by its corresponding design, as well as by the forward and inverse problems. The employed optimization algorithm, Cuckoo Search, is introduced in the next section. Subsequently, the procedure carried out is detailed in the Methodology section, and results achieved are presented and discussed afterwards. The manuscript wraps up by summarizing the main highlights and remarks.

1. The Heat Transfer Problem

The steady state heat transfer process of any thermal mechanism can be effectively approached through its equivalent thermal resistance (R [W/K]). This statement also covers a microchannel heat sink structure. Thus:

$$\dot{Q} = \theta/R, \tag{1}$$

where $\dot{Q}$ [W] is the net heat transfer rate entering into the system, and θ [ºC] is the finite difference of temperatures between the system isothermal boundary (T_i [ºC]) and its surroundings (T_a [ºC]) [13]. The latter is a measurable quantity allowing engineers to make decisions or control their systems. Specifically, MCHS performance is directly related to θ behavior in microelectronic thermal management applications, where electronic components must operate below a threshold temperature to avoid any failure. Thus, finite difference of temperatures between the chip–heat sink interface (T_i [ºC]) and the ambient (T_a [ºC]) can be written as,

$$\theta = T_i - T_a = QR. \tag{2}$$

Moreover, R is comprised of several heat transfer mechanisms within the MCHS, according to literature. In this study, two simple components are employed:

$$R = \left[2hNL_{hs}\left(w_c + \eta_p H_c\right)\right]^{-1} + \left[\rho_{wf} G_{wf} c_{wf}\right]^{-1}, \tag{3}$$

where the right hand terms of eq. (3) correspond to resistances due to convection inside channels, and to calorific capacitance of the working fluid, respectively. Hence, it is easy to notice that R depends on three kinds of parameters: design specifications, thermophysical properties and empirical correlations. The first set mainly consists on geometrical parameters, such as W_{hs} [m] and L_{hs} [m]. They respectively represent the width and length of the heat sink. H_c [m], $W_c = 2w_c$ [m], $W_p = 2w_p$ [m], and $N = (W_{hs} - W_c)/\left(W_c + W_p\right)$ are the channel height, channel width, wall width, and number of channels. This group also includes the non–geometrical design parameter G_{wf} [m^3/s], which is the volume flow rate of the working fluid. The second group contains thermophysical properties of the body material of the heat sink (hs) and working fluid (wf): density (ρ [kg/m^3]), thermal conductivity (k [W/m·K]), and specific heat capacity (c [J/kg·K]). The last classification of parameters corresponds to empirical correlations and related expressions, such as the convective heat transfer coefficient

(h[W/m^2·K]), and the efficiency (η_p). These parameters have a commonly accepted form, as:

$$h = \left(\frac{k_{wf}}{D_h}\right) Nu, \qquad \text{with } D_h = \frac{2W_c H_c}{W_c + H_c}, \tag{4}$$

$$\eta_p = \frac{\tanh(mH_c)}{mH_c}, \qquad \text{with } m = \sqrt{\frac{2h\left(w_p + L_{hs}\right)}{k w_p L_{hs}}}, \tag{5}$$

where Nu is the dimensionless Nusselt number, and D_h [m] is the effective hydraulic diameter of one channel.

Once design specifications, thermophysical properties and correlations are defined, the temperature difference in a heat sink application, θ, can be approximated via eq. (2). Let $\vec{x}$ be a vector containing all parameters required to obtain θ, $\theta(\vec{x}) : \boldsymbol{R}^D \to \boldsymbol{R}$ since $\vec{x} \in \boldsymbol{R^D}$. There are infinite possible values for $\vec{x}$, but their selection depends on the practical application and engineer expertise. Say $\vec{x}$ is formed by two parameter vectors, $\vec{x} = (\vec{y}, \vec{z})$, where $\vec{y}$ is the set of known parameters from a practical setup specification, *e.g.* size constraints, fluid flow pumping power and net heat flux. Likewise, $\vec{z}$ is the vector of parameters subject to the expert knowledge, whose values can enhance or jeopardize the system performance. As an illustrative example, $\vec{y}$ and $\vec{z}$ could be:

$$\vec{y} = \left(W_{hs}, L_{hs}, H_c \rho_{hs}, \rho_{wf}, k_{hs}, k_{wf}, c_{wf}, \dots\right)^\intercal,$$
$$\vec{z} = \left(w_c, w_p, G_{wf}\right)^\intercal,$$

Nevertheless, it is possible to reduce any uncertainty by finding, objectively, the best configuration for $\vec{z}$. Such configuration can be reached through several conceptual schemes. One example is to use a minimization procedure of the equivalent thermal resistance. Fortunately, there exists a recurrent and powerful methodology for finding these parameters, based on the second law of thermodynamics. This process is based on the Entropy Generation Minimization (EGM) criterion. EGM has been employed by several authors since 2009 for designing microchannel heat sinks [5, 13]. They minimize the total entropy generation rate ($\dot{S}_{gen}$ [W/K]), given by,

$$\dot{S}_{gen}(\vec{z}) = \left[\left(\frac{\dot{Q}^2}{T_a}\right)\frac{R(\vec{x})}{T_i(\vec{x})} + \left(\frac{G_{wf}}{T_a}\right)\Delta P(\vec{x})\right]_{\vec{y}}, \tag{6}$$

where T_a and T_i are temperatures of the ambient and the chip–heat sink interface, respectively. They are both related with θ as $\theta = T_i - T_a$, and thus with $\dot{Q}$ and R by eq. (1). Moreover, ΔP [Pa] is the pressure drop perceived by the working fluid flowing throughout the system with a volume flow rate of G_{wf} [m^3/s], which can be modelled as shown,

$$\Delta P = \frac{\rho_{wf}}{2}\zeta_1^2\left(f_c\frac{L_{hs}}{D_h} + 1.79 - 2.32\zeta_2 + 0.53\zeta_2^2\right), \tag{7}$$

$$\zeta_1 = \frac{G_{wf}}{NW_c H_c}, \qquad \text{and} \quad \zeta_2 = \frac{W_c}{W_c + W_p}. \tag{8}$$

In eq. (7), f_c is the Darcy friction factor for the inner walls of the channels. For the sake of brevity, more information about all described concepts and formulae can be found in [13]. However, some parameters from the third classification, *i.e.*, empirical correlations, are just approximate forms obtained for general cases which could be erroneous in several specific and suitable applications. Hence, a more accurate form or value for these parameters is needed.

Many studies have dealt with the estimation of unknown quantities, which usually comes from the solution of an inverse problem. They have used measurements from an experimental setup implementing the traditional Least Squared Error (LSE) criterion, which consists on minimizing $\varepsilon(\vec{z})$ [15], such as

$$\varepsilon(\vec{z}) = \left|\vec{\theta}_m - \theta(\vec{x})\right|_2^2, \tag{9}$$

since $\vec{\theta}_m$ [ºC] is the vector of temperature difference measurements at the chip–heat sink interface with respect the ambient. $\theta(\vec{x})$ [ºC] is the vector of values calculated for the same temperature difference, using a known functional form (or model), a set of known parameters $\vec{y}$, and candidate values for the unknown parameters $\vec{z}$, with $\vec{x} = (\vec{y}, \vec{z})$. $|\cdot|_2^2$ is the ordinary Euclidian two–norm.

Finally, all the aforementioned about the electronic thermal management scenario, based on a microchannel heat sink, can be summarized in three heat transfer problems presented as is now described.

1.1. Design heat transfer problem

In a given practical scenario for a microelectronic thermal management problem (*i.e.*, the conceptual design of a microchannel heat sink), with a defined set of specifications and constraints $\vec{y}$, it is possible to determine a set of parameters $\vec{z}$. They optimize the system performance, operating in steady state, in the sense of the entropy generation. From eq. (6),

$$\begin{aligned} \vec{z}^* &= \operatorname{argmin}_{\vec{z}}\{\dot{S}_{gen}(\vec{x})\}, \\ \text{s.t. } & \vec{z}_l \preccurlyeq \vec{z} \preccurlyeq \vec{z}_u. \end{aligned} \tag{10}$$

1.2 Objective of the direct heat transfer problem

The direct heat transfer problem is laid out with the objective of calculating the temperature in a chip–heat sink interface, for a specific microelectronic heat sinking process. Heat power dissipation is assumed to be uniform and homogeneous. Other known parameters and constrictions are given by $\vec{x} = (\vec{y}, \vec{z})$. They are evaluated using the mathematical model from eq. (2), as

$$\theta(\vec{x}) \triangleq \dot{Q}R(\vec{x}), \tag{11}$$

where $\vec{x} = \left(W_{hs}, L_{hs}, H_c, w_c, w_p, \dots\right)^\intercal$ is the vector of all parameters detailed in eq. (3).

Some parameters of $\vec{x}$ can be determined via a design process for a given set of specifications and constraints $\vec{y}$, as it was mentioned in the design heat transfer problem.

1.3 Objective of the inverse heat transfer problem

In contrast, the inverse heat transfer problem is laid out with the objective of estimating some parameter values of $\vec{x}$ which are unrelated to the design specifications $\vec{y}$, say $\vec{z}$ as $\vec{x} = (\vec{y}, \vec{z})$. The selected scenario for this corresponds to a practical microchannel heat sink, where at least one temperature is measurable. This can be achieved by solving

$$\begin{aligned} \vec{z}^* &= \operatorname{argmin}_{\vec{z}}\{\varepsilon(\vec{x})\}, \\ \text{s.t. } & \vec{z}_l \preccurlyeq \vec{z} \preccurlyeq \vec{z}_u. \end{aligned} \tag{12}$$

where $\vec{y}$ and $\vec{z}$ are vectors of known and unknown parameters, respectively, since $\vec{x} = (\vec{y}, \vec{z})$ is the vector of all parameters from the direct problem formulation, that is $\vec{x} = \left(W_{hs}, L_{hs}, H_c, w_c, w_p, \dots\right)^\intercal$. The set of values of $\vec{z}$ for the experimental setup can be obtained from a conceptual design, or from previous knowledge of the direct problem solution.

2. Cuckoo Search algorithm

Cuckoo Search (CS) is a modern global optimization algorithm. CS has been widely implemented. It was formulated by Yang and Deb in 2009, as a bio-inspired technique that mimics the brood parasitism behavior of certain species of cuckoos in nature [32]. CS can be described in few words as a mutation–based swarm–intelligence algorithm with Lévy flights. It is formally described by using the following mathematical definitions, and its overall logic is laid out in Pseudocode 1.

Definition 1. Let $\mathfrak{X}^t = \{\vec{x}_1^t, \dots, \vec{x}_N^t\}$ be a finite set of candidate solutions for any optimization problem in $\boldsymbol{R^D}$, with known cost function $f\colon \boldsymbol{R^D} \to \boldsymbol{R}$. N is the number of candidate solutions, D is the number of unknown variables and, $\vec{x}_k^t \in R^D$ denotes the k–th candidate solution at the step t of an iterative procedure, let it be Cuckoo Search algorithm.

Definition 2. Let $\mathfrak{X}^{t+1}$ represents the finite set of new candidate solutions which improves the previous set of solutions $\mathfrak{X}^t$ through two strategies, *i.e.*, Lévy flights and eggs discovery. Each candidate of $\mathfrak{X}^{t+1}$ suffices $f(\mathfrak{X}^{t+1}) \leq f(\mathfrak{X}^t)$.

Definition 3. Let $\vec{x}_*^t \in \mathfrak{X}^t$ be the best solution found at the t–th iteration, determined by $\vec{x}_*^t = \operatorname{argmin}\{f(\mathfrak{X}^t)\}$.

Definition 4. Strategy 1—Let $\vec{x}_k^{t+1} \in \boldsymbol{R^D}$ be a new candidate position obtained by

$$\vec{x}_k^{t+1} = \vec{x}_k^t + \delta_x \vec{\lambda}_L \odot (\vec{x}_k^t - \vec{x}_*^t), \tag{13}$$

where δ_x is the step size, commonly set as 0.1 for $D \leq 3$ and 0.01 for $D > 3$. $\vec{\lambda}_L$ is a vector of i.i.d. symmetric Lévy stable random numbers, and $\odot$ is the Hadamard–Schur product.

Definition 5. Strategy 2—Let $\vec{x}_k^{t+1} \in \mathbf{R}^{\boldsymbol{D}}$ be a new candidate position with an associated probability p_D, related to the chance that a host bird discovers a hidden egg, which is determined as

$$\vec{x}_k^{t+1} = \vec{x}_k^t + \vec{u} \odot \hbar(\vec{u} - p_D) \odot \left(\vec{x}_i^t - \vec{x}_j^t\right), \tag{14}$$

where $\vec{u} \in \boldsymbol{R^D}$ is a vector of i.i.d. uniform random numbers between 0 and 1, $\hbar$ is a multidimensional form of the Heaviside function, and $\vec{x}_i^t, \vec{x}_j^t \in \mathfrak{X}^t$ with $i \neq j$, where i and j are selected randomly.

```
Pseudocode 1.
Cuckoo Search (CS) algorithm
INPUT: $f\colon \boldsymbol{R^D} \to \boldsymbol{R}$, $N > 2$, $\delta_x$, $p_D \in [0,1]$, and stopping criteria: $M \gg 1$,
  and others (if they exist)
OUTPUT: $\vec{x}_*^t$ from Definition 3.
  STEP 1: Make $t \leftarrow 0$ and initialize $\mathfrak{X}^0$ using Definition 1
  STEP 2: Find $\vec{x}_*^0$ using Definition 3
WHILE ($t \leq M$) AND (any stopping criteria is not reached) DO
  STEP 3: Update $\mathfrak{X}^{t+1}$ according to Definition 2 using Definition 4
  STEP 4: Update $\mathfrak{X}^{t+1}$ according to Definition 2 using Definition 5
  STEP 5: Find $\vec{x}_*^{t+1}$ using Definition 3, and make $t \leftarrow t + 1$
END WHILE
```

3. Methodology

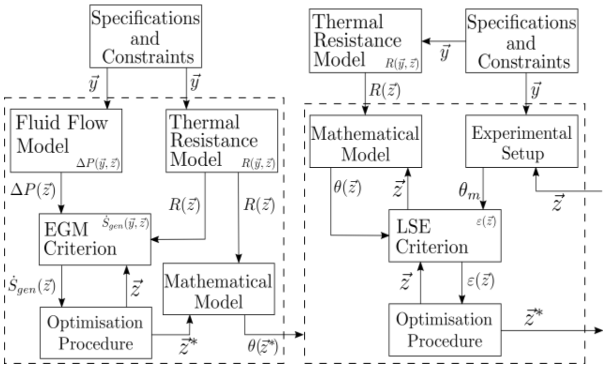
All experiments were performed in a numerical computing platform, running on an iMac model 15.1, with an Intel Core i5 CPU at 1.6–2.7 GHz, 8 GB RAM, and macOS Sierra v10.12.1. Each case of study was repeated a hundred times for statistical purposes. As previously stated, we used the Cuckoo Search (CS) algorithm as an optimizer. Parameters for CS were tuned following [33].

This work was carried out in two stages, which are graphically presented in Fig. 1. System specifications are laid out in Table 1. Moreover, two materials for the heat sink body, and two fluids for the coolant were considered, as Table 2 shows. In the first case we used Silicon (Si) and High Thermal Conductive Graphite (HTCG). In the second case, we used air and ammonia gas (NH3). Fig. 1a shows the solution of the forward problem, equation (11), under several design conditions and specifications ($\vec{y}$), and with multiple target net heat power ($\dot{Q}$ [W]) values. Design parameters ($\vec{z}$) were obtained by minimising the entropy generation rate of the entire system, *i.e.*, by solving eq. (6) with the Cuckoo Search algorithm presented in Pseudocode 1. Parameters h and f_c from eq. (4) and (7), respectively, were calculated using equations (15) – (17), where Re is the Reynolds number [13].

$$h = \left(\frac{k_{wf}}{D_h}\right)\left[2.253 + 8.164\left(1 + \frac{W_c}{H_c}\right)^{-1.5}\right], \tag{15}$$

$$f_c = Re^{-1}(\zeta_3^{1.14} + \zeta_4^2)^{0.5}, \tag{16}$$

$$\zeta_3 = 7.6953 Re \frac{D_h}{L_{hs}}, \zeta_4 = 24.34 - 9.82 \frac{D_h}{W_c + H_c}, \tag{17}$$



(a) (b)

Figure 1. (a) Direct and (b) Inverse Heat Transfer Problems.

Table 1.
Values assumed for design parameters of the system.

Design specifications, $\vec{y}$		Design variables, $\vec{z}$	
Parameter	Value	Parameter	Value
H_c	1.7 mm	G_{wf}	(0, 0.01] m^3/s
L_{hs}, W_{hs}	51 mm	$\alpha = W_c/H_c$	(0, 1]
T_a	300 K	$\beta = W_c/W_p$	[1, 100]
$\dot{Q}$	25, 30, ... , 150 W		

Table 2.
Thermophysical property values for heat sink materials and working fluids: High Thermal Conductive Graphite (HTCG), Silicon (Si), Air, and Ammonia Gas (NH$_3$).

Property	HTCG	Si	Air	NH$_3$
ρ [kg/m^3]	1000	2330	1.1614	0.7
k [W/m·K]	1900	148	0.0261	0.0270
c [J/kg·K]	742	703	1007	2158
$\nu \times 10^{-5}$ [m^2/s]	-	-	1.58	1.47

4. Results and discussion

Fig. 2 shows the optimal values of design parameters G^*_{wf}, α^* and β^*. This means that $\dot{S}_{gen,min} = \dot{S}_{gen}(\alpha^*, \beta^*, G^*_{wf})$. Or, in other words, that such parameters minimize entropy production of the microchannel heat sink. Each one of the plotted markers correspond to an optimal design for all combinations of build materials, such as silicon (Si) and High Thermal Conductive Graphite, and working fluids like Air and Ammonia gas (NH$_3$).

Likewise, Fig. 3 presents the minimal entropy generation rate ($\dot{S}_{gen,min}$) for each one of the obtained designs. Values are plotted for different electronic power dissipation ($\dot{Q}$) levels. As observed, the entropy generation rate increases (as expected) as the net heat power is augmented. There is also a noticeable difference when choosing a different material (i.e. Si or HTCG) to build the MCHS. This effect is also evident when choosing a different fluid (*i.e.*, Air or NH$_3$) to act as a coolant. The combination HTCG-NH$_3$ laid out best results in terms of minimal entropy production, overcoming the traditional combination of Si-Air, which corroborates reported results in the literature. Moreover, Fig. 3 evidences the influence of implementing a certain material or fluid. For high power applications, *i.e.*, beyond 60 W, this effect becomes more evident.

This influence is strongly reflected on the behavior of the optimal equivalent thermal resistance (R^*). Figure 4 shows data for each fluid. In this figure, R^* decreases when $\dot{Q}$ increases to avoid an excessive rise of temperature inside the electronic package, and reducing the overall effects due to irreversibilities in the MCHS.

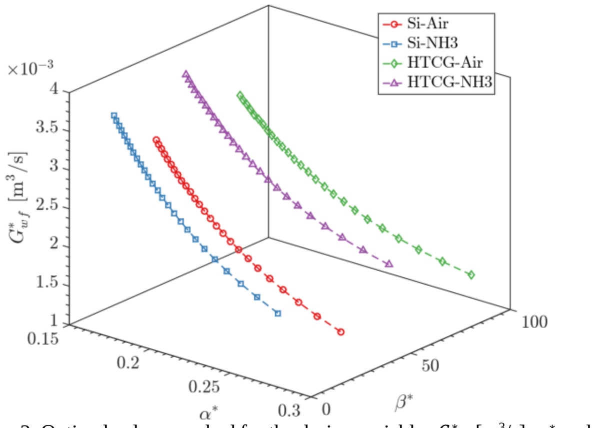


Figure 2. Optimal values reached for the design variables G^*_{wf} [m^3/s], α^* and β^* varying the net heat power dissipation ($\dot{Q}$ [W]) from an electronic device.

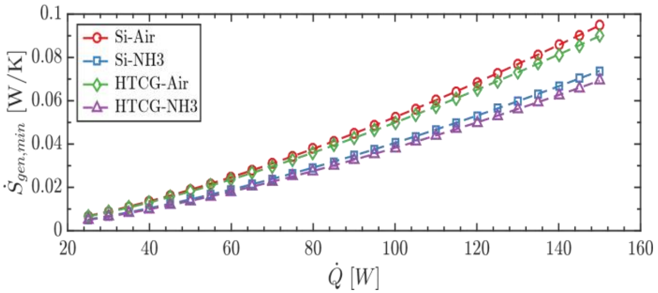


Figure 3. Minimal entropy generation rate ($\dot{S}_{gen,min}$ [W/K]) varying the net heat power dissipation ($\dot{Q}$ [W]) from an electronic device.

Furthermore, optimal difference temperature values of θ^* can be calculated from the R^* data (Fig. 4), using eq. (2). These values are presented in Fig. 5. Such information complements the minimal entropy production reached from the design procedure. Besides, it is a practical and measurable quantity, which describes the performance of the thermal management system.

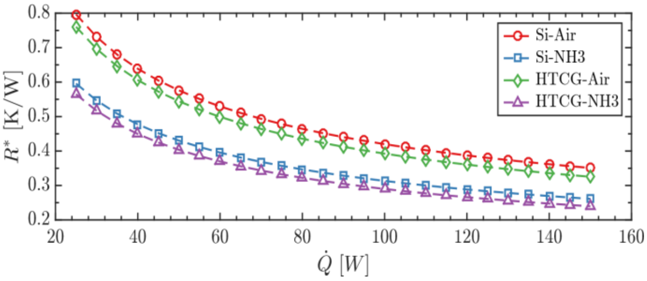


Figure 4. Optimal values of equivalent thermal resistance (R^* [K/W]) varying the net heat power dissipation ($\dot{Q}$ [W]) from an electronic device.

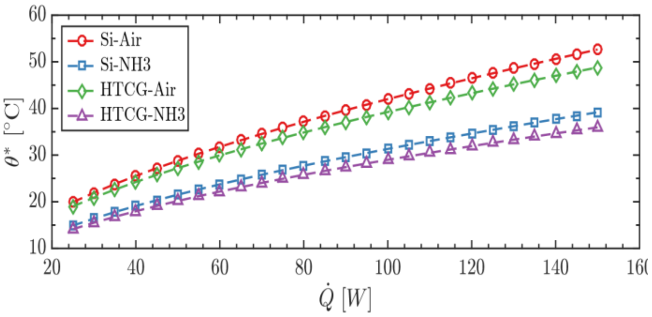


Figure 5. Optimal values of temperature finite difference (θ^* [ºC]) varying the net heat power dissipation ($\dot{Q}$ [W]) from an electronic device.

Fig. 6 displays results from the forward problem, in terms of θ, contaminated with additive white Gaussian noise (AWGN), to emulate a measured dataset. We employed several values of signal–to–noise ratio (SNR [dB]) to analyze the performance of our approach under different scenarios. The measured temperature difference is identified as θ_m.

Fig. 7 presents results from the inverse heat transfer problem, where θ_e is the estimated temperature difference. Furthermore, estimated convection heat transfer coefficients are shown in Fig. 8. Striving to complement our data, Table 3 summarizes the root-mean-square error (RMSE) for each combination of material and fluid, and for all noise levels.

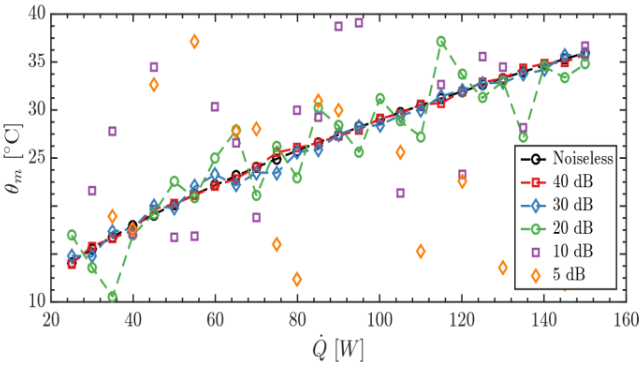


Figure 6. Measured temperature difference (θ_m [ºC]) values, varying the net heat power dissipation ($\dot{Q}$ [W]), with several values of noise (SNR [dB])

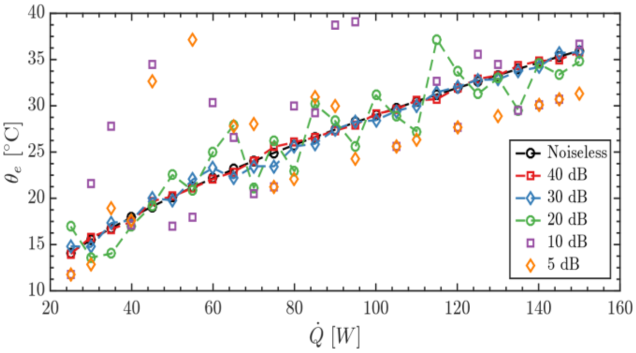


Figure 7. Estimated temperature difference (θ_e [ºC]) values, varying the net heat power dissipation ($\dot{Q}$ [W]), with several values of noise (SNR [dB])

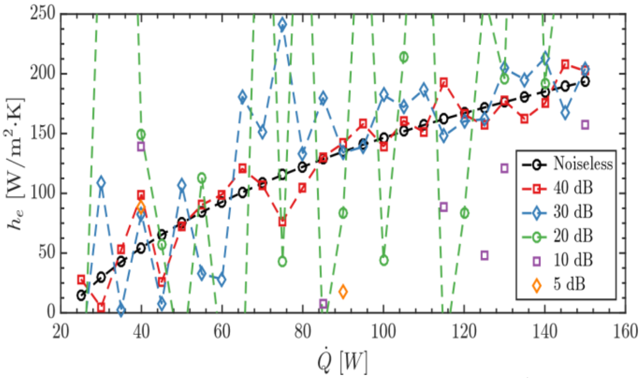


Figure 8. Estimated film heat transfer coefficient (h_e [W/m^2·K]) values, varying the net heat power dissipation ($\dot{Q}$ [W]), with several values of noise (SNR [dB])

Table 3.
Root-mean-square error (RMSE) of estimated values for h, varying the net heat power $\dot{Q}$, with different materials and working fluids.

	RMSE			
SNR [dB]	Si-Air	Si-NH$_3$	HTCG-Air	HTCG-NH$_3$
Noiseless	0.00	0.00	0.00	0.00
40	2.04	1.64	0.27	0.34
30	3.17	7.52	0.70	0.99
20	8.55	72.22	5.79	4.39
10	12.48	27.54	5.33	10.00
5	19.28	33.78	6.22	10.70

5. Conclusions

This article proposed an alternative strategy for estimating convection heat transfer coefficients in electronic thermal management applications. Such a strategy solves the associated inverse heat transfer problem. In the present case, we tackled this inverse problem via the Cuckoo Search (CS) algorithm, by minimizing a cost function based on the least-square error. This methodology was illustrated with a microchannel heat sink with several design conditions. Average convective heat transfer coefficients were estimated from several synthetical temperature measurements, with different noise levels. Reference temperature data were obtained by solving the forward problem, based on the equivalent thermal resistance model of a previously designed microchannel. Finally, these designs were achieved through the entropy generation minimization criterion, also powered by CS, employing several specifications (material, working fluid and $\dot{Q}$). Results agreed remarkably well for signal-to-noise ratios above 30 dB, which can be reached with any modern temperature sensors.

Acknowledgement

The authors gratefully acknowledge the support of the *Consejo Nacional de Ciencia y Tecnología* (CONACyT) of Mexico, under grant number 578594, and of the Universidad Industrial de Santander, Colombia.

References

[1] Yun, B., Shin, K. G., and Wang, S. Predicting thermal behavior for temperature management in time-critical multicore systems, Real-Time Technology and Applications - Proceedings, 2013. pp. 185-194. doi:10.1109/RTAS.2013.6531091.

[2] Khaleduzzaman, S., Sohel, M., Saidur, R., Mahbubul, I., Shahrul, I., Akash, B., and Selvaraj, J. Energy and exergy analysis of alumina water nanofluid for an electronic liquid cooling system. International Communications in Heat and Mass Transfer, 57, pp. 118-127, 2014. doi:10.1016/j.icheatmasstransfer.2014.07.015.

[3] Kandlikar, S. G. Review and Projections of Integrated Cooling Systems for Three-Dimensional Integrated Circuits. Journal of Electronic Packaging, 136 (2) p. 24001, 2014. doi:10.1115/1.4027175

[4] Chen, X., Ye, H., Fan, X., Ren, T., and Zhang, G. A review of small heat pipes for electronics. Applied Thermal Engineering, 96, pp. 1-17, 2016. doi:10.1016/j.applthermaleng.2015.11.048.

[5] Alfaryjat, A. A., Stanciu, D., Dobrovicescu, A., Badescu, V., and Aldhaidhawi, M. Numerical investigation of entropy generation in microchannels heat sink with different shapes. IOP Conference Series: Materials Science and Engineering, 147, p. 012134, 2016, doi:10.1088/1757-899X/147/1/012134.

[6] Sahoo, S. K., Das, M. K., and Rath, P. Application of TCE-PCM based heat sinks for cooling of electronic components: A review. Renewable and Sustainable Energy Reviews, 59, pp. 550-582, 2016. doi:10.1016/j.rser.2015.12.238.

[7] Kadam, S. T., and Kumar, R. Twenty first century cooling solution: Microchannel heat sinks. International Journal of Thermal Sciences, 85, pp. 73-92, 2014. doi:10.1016/j.ijthermalsci.2014.06.013.

[8] Lin, L., Chen, Y.-Y., Zhang, X.-X., and Wang, X.-D. Optimization of geometry and flow rate distribution for double-layer microchannel heat sink. International Journal of Thermal Sciences, 78, pp. 158-168, 2014. doi:10.1016/j.ijthermalsci.2013.12.009.

[9] Yang, Y.-T., Tsai, K.-T., Wang, Y.-H., and Lin, S.-H. Numerical study of microchannel heat sink performance using nanofluids. International Communications in Heat and Mass Transfer, 57, pp. 27-35, 2014. doi:10.1016/j.icheatmasstransfer.2014.07.006.

[10] Tuckerman, D. B., and Pease, R. F. W. High-performance heat sinking for VLSI. IEEE Electron Device Letters, 2 (5), pp. 126-129, 1981. doi:10.1109/EDL.1981.25367.

[11] Shao, B., Sun, Z., and Wang, L. Optimization design of microchannel cooling heat sink. International Journal of Numerical Methods for Heat & Fluid Flow, 17 (6), pp. 628-637, 2007. doi:10.1108/09615530710761243.

[12] Chen, Y., Peng, B., Hao, X., and Xie, G. Fast approach of Pareto-optimal solution recommendation to multi-objective optimal design of serpentinechannel heat sink. Applied Thermal Engineering, 70 (1), pp. 263-273, 2014. doi:10.1016/j.applthermaleng.2014.05.004.

[13] Khan, W. A., Culham, J. R., and Yovanovich, M. M. Optimization of Microchannel Heat Sinks Using Entropy Generation Minimization Method. IEEE Transactions on Components and Packaging Technologies, 32 (2), pp. 243-251, 2009. doi:10.1109/TCAPT.2009.2022586.

[14] Bejan, A. Entropy generation minimization: The new thermodynamics of finite size devices and finite-time processes. Journal of Applied Physics, 79 (3), pp. 1191-1218, 1996.

[15] Ozisik, M. N., Inverse heat transfer: fundamentals and applications. CRC Press, 2000.

[16] Wang, G., Zhang, L., Wang, X., and Tai, B. L. An inverse method to reconstruct the heat flux produced by bone grinding tools. International Journal of Thermal Sciences, 101, pp. 85-92, 2016. doi:10.1016/j.ijthermalsci.2015.10.021.

[17] Li, J., Jiang, N., Gao, Z., Liu, H., and Wang, G. An inverse heat conduction problem of estimating the multiple heat sources for mould heating system of the injection machine. Inverse Problems in Science and Engineering, 24 (9), pp. 1587-1605, 2016. doi:10.1080/17415977.2015.1130044.

[18] Cornejo, I., Cornejo, G., Ramírez, C., Almonacid, S., and Simpson, R. Inverse method for the simultaneous estimation of the thermophysical properties of foods at freezing temperatures. Journal of Food Engineering, 191, pp. 37-47, 2016. doi:10.1016/j.jfoodeng.2016.07.003.

[19] Wang, Y., Luo, X., Yu, Y., and Yin, Q. Evaluation of heat transfer coefficients in continuous casting under large disturbance by weighted least squares Levenberg-Marquardt method. Applied Thermal Engineering, 111, pp. 989-996, 2017. doi:10.1016/j.applthermaleng.2016.09.154.

[20] Mohebbi F, Sellier, M., and Rabczuk, T. Estimation of linearly temperature-dependent thermal conductivity using an inverse analysis. International Journal of Thermal Sciences, 117, pp. 68-76, 2017. http://dx.doi.org/10.1016/j.ijthermalsci.2017.03.016.

[21] Luo, X., and Yang, Z. A new approach for estimation of total heat exchange factor in reheating furnace by solving an inverse heat conduction problem. International Journal of Heat and Mass Transfer, 112, pp. 1062-1071, 2017.doi:101016/j.ijheatmasstransfer201705009

[22] Chanda, S., Balaji, C., Venkateshan, S. P., and Yenni, G. R. Estimation of principal thermal conductivities of layered honeycomb composites using ANNGA based inverse technique. International Journal of Thermal Sciences, 111, pp. 423-436, 2017. doi:10.1016/j.ijthermalsci.2016.09.011.

[23] Dhiman, J., and Prasad, S.K. Inverse estimation of heat flux from a hollow cylinder in cross-flow of air. Applied Thermal Engineering, 113, pp. 952-961, 2017. doi:101016/j.applthermaleng.201611088.

[24] Mu, S., Li, H., Wang, J., and Liu, X. Optimization based inversion method for the inverse heat conduction problems. IOP Conference Series: Earth and Environmental Science, 64 (1), p. 9, 2017. doi:10.1088/1755-1315/64/1/012094.

[25] Duda, P. Solution of inverse heat conduction problem using the Tikhonov regularization method. Journal of Thermal Science, 26 (1) pp. 60-65, 2017. doi:10.1007/s11630-017-0910-2.

[26] Bodla, K. K., Murthy, J. Y., and Garimella, S. V. Optimization Under Uncertainty for Electronics Cooling Design Applications, 13th IEEE ITHERM Conference, 2012. pp. 1191-1201.

[27] Bodla, K. K., Murthy, J. Y., and Garimella, S. V. Optimization Under Uncertainty Applied to Heat Sink Design. Journal of Heat Transfer, 135 (1), p. 011012, 2012. doi:10.1115/1.4007669.

[28] Huang, C. H., Liu, Y. C., and Ay, H. The design of optimum perforation diameters for pin fin array for heat transfer enhancement. International Journal of Heat and Mass Transfer, 84, pp. 752-765, 2015. doi:10.1016/j.ijheatmasstransfer.2014.12.065.

[29] Sudheesh, R. S., and Prasad, N. S. Comparative Study of Heat Transfer Parameter Estimation Using Inverse Heat Transfer Models of a Trailing Liquid Nitrogen Jet in Welding. Heat Transfer Engineering, 36 (2), pp. 178-185, 2015. doi:10.1080/01457632.2014.909219.

[30] Chen, H.-T., and Wu, C.-Y. Estimate of Mixed Convection Heat Transfer and Fluid Flow Characteristics of Plate-fin Heat Sinks in the Opening of the Enclosure. Universal Journal of Mechanical

Engineering, 4 (5), pp. 107-112, 2016. doi:10.13189/ujme.2016040501.

[31] Chen, H.-T., Tseng, H.-C., Jhu, S.-W., and Chang, J.-R. Numerical and experimental study of mixed convection heat transfer and fluid flow characteristics of plate-fin heat sinks. International Journal of Heat and Mass Transfer, 111, pp. 1050-1062, 2017. doi:10.1016/j.ijheatmasstransfer.2017.04.065.

[32] Yang, X.-S., and Deb, S. Engineering Optimisation by Cuckoo Search, Int. J. Mathematical Modelling and Numerical Optimisation, 1 (4), pp. 330-343, 2010. arXiv:1005.2908, doi:10.1504/IJMMNO.2010.035430.

[33] Cruz-Duarte, J. M., Garcia-Perez, A., Amaya-Contreras, I. M., Correa-Cely, C. R., Romero-Troncoso, R. J., and Avina-Cervantes, J. G. Design of Microelectronic Cooling Systems Using a Thermodynamic Optimization Strategy Based on Cuckoo Search. IEEE Transactions on Components, Packaging and Manufacturing Technology, 99, pp. 1-9, 2017. doi:10.1109/TCPMT.2017.2706305.

Advanced portfolio optimization problems

T. Tichý[1], A. Kresta[1], S. Ortobelli[1], N. Kouaissah[1]

1. VSB-TU Ostrava, Sokolska tr. 33, 702 00 Ostrava
tomas.tichy@vsb.cz

Keywords: portfolio optimization, risk measures, dependency measures, large scale portfolio problem, optimization algorithm

1 Introduction

In this work potential usage and impact of various algorithms of metaheuristic specification for portfolio optimization are studied. In particular, we focus on problems, in which different risk and dependency measures are involved and their non-parametric estimation is required. For this purposes, novel approach based on fuzzy transform approach is utilized. Particular approaches are evaluated using real data from various markets.

References

Barak, S., Dahooie, J.H., Tichý, T. Wrapper ANFIS-ICA method to do stock market timing and feature selection on the basis of Japanese Candlestick. *Expert Systems with Applications* **42** (23): 9221-9235, 2015.

Holčapek, M., Tichý, T. 2010. A probability density function estimation using F-transform, *Kybernetika*, vol. 46, no. 3, pp. 447– 458, 20.

Holčapek, M., Tichý, T. 2011 A smoothing filter based on fuzzy transform, *Fuzzy Sets Syst*., vol. 180, no. 1, pp. 69–97.

Holčapek, M., Tichý, T. 2015. Discrete Multivariate F-transform of Higher Degree. *IEEE International Conference on Fuzzy Systems*, 5p.

Ortobelli, S., Lando, T., Petronio, F., Tichý, T. Asymptotic Multivariate Dominance: a Financial Application. *Working paper submitted to Methodology and Computing in Applied Probability*.

Ortobelli, S., Lando, T., Petronio, F., Tichý, T. Asymptotic stochastic dominance rules for sums of i.i.d. random variables. *Journal of Computational and Applied Mathematics* 300, 432−448, 2016.

Ortobelli, S., Cassader, M., Vitali, S., Tichý, T. Portfolio selection strategy for the fixed income markets with immunization on average. *Annals of Operations Research.* To appear in 2017.

Ortobelli, S., Tichý, T. On the impact of semidenite positive correlation measures in portfolio theory. *Annals of Operations Research* 235: 625−652, 2015. ISSN 0254-5330. 10.1007/s10479-015-1962-x.

Tichý, T. *Lévy Processes in Finance: Selected applications with theoretical background*. SAEI, vol. 9. Ostrava: VŠB-TU Ostrava, 2011.

A Matheuristic for the Rainbow Cycle Cover problem

J. Moreno[1], S. Martins[2], and Y. Frota[3]

[1] Department of Computer Science/Fluminense Federal University, 24210-240, Brazil
jmoreno@ic.uff.br
[2] Department of Computer Science/Fluminense Federal University, 24210-240, Brazil
simone@ic.uff.br
[3] Department of Computer Science/Fluminense Federal University, 24210-240, Brazil
yuri@ic.uff.br

1 Introduction

Graphs are often used to model relationships between elements. In some situations, it is necessary to distinguish the different types of connections between the vertices of the graphs [1–4]. Such situations can be modeled by multicolored graphs. In this context, an edge-colored graph is a graph $G = (V, E)$, with a set of labels or colors L and a color function $f_c : L \rightarrow E$ which assigns a color from L to each edge of E.

A rainbow cycle [5] is a cycle with all its edges with different colors. Single vertices are considered trivial rainbow cycles. A rainbow cycle cover for the graph G is defined as a disjoint collection of rainbow cycles, which means that each vertex can only belong to exactly one rainbow cycle. The Rainbow Cycle Cover (RCC) problem consists of finding the minimum number of disjoint rainbow cycles covering G.

Figure 1 shows an example of an edge-colored graph with 5 possible colors (labels) for the edges. A feasible solution for the RCC problem on this graph is shown in Figure 2. In this case there are two non-trivial rainbow cycles and one trivial rainbow cycle.

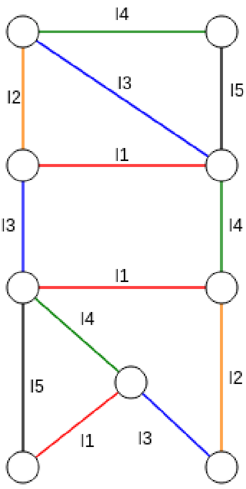


Fig. 1: A multicolored graph

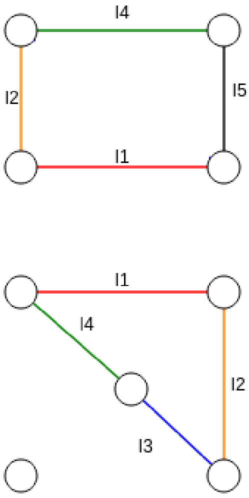


Fig. 2: A rainbow cycle cover of the graph with two non-trivial rainbow cycles and one trivial rainbow cycle.

The RCC problem has been proved to be NP-hard in [6]. In [7] it is shown that it is still NP-hard even if the graph does not have cycles of size 3. Recently, an integer mathematical formulation was proposed for this problem and tested for 3 hours in instances with up to 50 vertices. The experiments showed that this problem increases its complexity considerably as the number of vertices increases [8].

In the last years, different strategies have been developed that combine exact and heuristics algorithms. In this context, matheuristics [9–11] have attracted the attention of the scientific community. Matheuristics may use some features of the mathematical models developed for the problems to customize heuristics to solve these problems or use heuristics to improve time effectiveness of mathematical programming techniques.

In this work, we develop a matheuristic based on the Iterated Local Search metaheuristic for the RCC problem, in order to quickly obtain solutions with good quality. The remainder of this paper is organized as follows. Section 2 presents the mathematical formulation used for the RCC problem. In Section 3, the algorithms that compose the matheuristic are presented. In Section 4, experimental results obtained with the instances proposed in [12] and [13] are presented. Finally, in Section 5 the conclusions of this paper are discussed.

2 Mathematical Formulation

Given an undirected graph $G = (V, E)$, where V denotes the set of vertices ($|V| = n$) and E the set of edges, let $\delta(v)$ be the set of edges incident to a vertex v in G, where $d(v) = |\delta(v)|$ denotes the degree of vertex v. The set $E_l = \{e \in E \mid f_c(e) = l\}$ is the set of edges associated with a color $l \in L$. Finally, R denotes the family of all non-trivial rainbow cycles of the graph G, where a non-trivial rainbow cycle $H \in R$ is defined by its edge set, i.e., $H \subseteq E$.

A formulation for the RCC problem was used in [8]. This formulation uses a set of indices $I = \{0, \ldots, \bar{c} - 1\}$ that represents possible non-trivial cycles in a solution. Note that if this value is unknown, then it can be set to $\bar{c} = \lfloor \frac{|V|}{3} \rfloor$, that occurs when all rainbow cycles have the minimum length (3 edges).The formulation uses the binaries variables y_v^c (set to 1, if and only if the vertex v belongs to the non-trivial cycle c) and x_e^c (set to 1, if and only if the edge e belongs to the non-trivial cycle c). The formulation is shown below:

$$\min \sum_{c \in I} \gamma_c + \sum_{v \in V} \left(1 - \sum_{c \in I} y_v^c \right) \tag{1}$$

subject to:

$$\sum_{v \in V} y_v^c \leq |L| \gamma_c, \;\; c \in I \tag{2}$$

$$\sum_{v \in V} y_v^c \geq 3\gamma_c, \;\; c \in I \tag{3}$$

$$\sum_{c=0}^{\bar{c}-1} y_v^c \leq 1, \;\; v \in V \tag{4}$$

$$\sum_{e \in \delta(v)} x_e^c = 2y_v^c, \;\; v \in V, c \in I \tag{5}$$

$$\sum_{e \in H} x_e^c + x_f^c \leq |H|, H \in R, \;\; f \in E \setminus H, c \in I \tag{6}$$

$$\sum_{e \in E_l} x_e^c \leq 1, \;\; l \in L, c \in I \tag{7}$$

$$\gamma_{c+1} \leq \gamma_c, \;\; c \in \{0, \ldots, \bar{c} - 2\} \tag{8}$$

$$\sum_{c=v+1}^{\bar{c}} y_v^c = 0, \;\; v \in V : v < \bar{c} \tag{9}$$

$$y_v^c \leq \sum_{w<v} y_w^{c-1}, \;\; v \in V \setminus \{0\}, c \in \{2, \ldots, \bar{c} - 1\} \tag{10}$$

where $\delta(S) = \{\{u, v\} \in E \mid u \in S \text{ and } v \in V \backslash S\}$.

The objective function (1) aims to minimize the number of rainbow cycles. The first part minimizes the sum of the number of non-trivial cycles γ_c. The second part tries to force the vertices to belong to a cycle, giving a sufficient large weight M to each trivial cycle (i.e. isolated vertex).

Constraints (2) express that the number of vertices belonging to a non-trivial cycle cannot be greater than the number of colors. Constraints (3) ensure that a non-trivial cycle must have at least three vertices. Constraints (4) guarantee that a vertex will belong to at most one non-trivial cycle. Constraints (5) ensure that a vertex belonging to a non-trivial cycle must have exactly two adjacent edges. Moreover, constraints (6) prevent two different non-trivial cycles to use the same variable γ_c. Constraints (7) impose that in each non-trivial cycle there may be at most one edge with color l.

Constraints (8)-(10) are not necessary for the model, but their inclusion helps to eliminate symmetric solutions. Constraints (8) ensure that a variable for the non-trivial cycle γ_c is not used if the variable γ_{c-1} is not used. Constraints (9) indicate that a vertex v, with index lower than $\overline{c}$, cannot belong to a non-trivial cycle with index c, such that $c > v$. Finally constraints (10) impose that a vertex v can only belong to a non-trivial cycle with index c if at least one vertex w, where $w < v$, belongs to the non-trivial cycle with index $c - 1$.

3 Matheuristic for the RCC problem

Algorithm 1 shows the pseudocode for the ILS heuristic [14] developed to quickly obtain good quality solutions for the RCC problem. First, an initial solution is generated for the problem (line 1) and a local search is applied to this solution to improve the quality of the constructed solution (line 2). Between lines 3 and 7, iterations are performed until a stopping criterion is reached. In each iteration, a perturbation in the current solution is done trying to escape of a local optimum and then a local search is performed. In line 6, a criterion is used to decide if the current solution will be replaced by the new generated solution. In our implementation, the best solution found is updated every time a better solution is obtained in the local search. Moreover, we use as stop condition that a maximum of 10 iterations be performed, as we want to quickly get a solution.

Algorithm 1: Iterated Local Search

```
  Input: The graph $G$.
  Output: A rainbow cycle cover $R$.
1 $x_0 \leftarrow$ Initial_Solution($G$)
2 $x^* \leftarrow$ Local_Search($G$)
3 repeat
4     $x' \leftarrow$ Perturbation($x^*$)
5     $x^{*'} \leftarrow$ Local_Search($x'$)
6     $x^* \leftarrow$ AcceptanceCriterion($x^*, x^{*'}, history$)
7 until termination condition met
8 return $x^*$
```

3.1 Initial Solution

The mathematical formulation for the RCC problem works well for small graphs, but as the size of the graphs increases, the computational time to solve the problem increases very fast. One of the causes is the large number of variables and restrictions of this model.

Let $ILP^{(q)}(G)$ be the procedure that solves the formulation for the RCC in the graph G, restricting to q the number of non-trivial cycles. Algorithm 2 shows the pseudocode used to construct an initial solution for the RCC problem.

Basically, Algorithm 2 solves the mathematical formulation by restricting the maximum number of non-trivial cycles to two. Each non-trivial cycle found will be part of the solution under construction. Then, the algorithm solves again the restricted mathematical formulation for the induced graph on the set of isolated vertices. This process is repeated until the solution for the induced graph does not contain any non-trivial cycle. In that case, all the vertices will be added to the solution as trivial cycles (line 11). When fixing the maximum number of non-trivial cycles for the formulation, the number of the variables and the restrictions are considerably reduced.

Algorithm 2: Initial Solution

```
Input: The graph $G$.
Output: A rainbow cycle cover $R$.
$R \leftarrow \emptyset$
$R' \leftarrow ILP^{(2)}(G)$
$N \leftarrow$ non-trivial cycles in $R'$
$T \leftarrow$ trivial cycles in $R'$
while $N \neq \emptyset$ do
    $R \leftarrow R \cup N$
    $H \leftarrow$ induced graph by $T$
    $R' \leftarrow ILP^{(2)}(H)$
    $N \leftarrow$ non-trivial cycles in $R'$
    $T \leftarrow$ trivial cycles in $R'$
$R \leftarrow R \cup T$
return $R$
```

3.2 Local Search

Once an initial solution is obtained, a local search is applied (Algorithm 3). A neighbor solution of a current solution is obtained as follow. First, any two non-trivial cycles are removed and their vertices are added to the set of isolated vertices. The other non-trivial cycles are fixed. Then, $ILP^{(q)}(G)$ is applied over the set of isolated vertices limiting the maximum number of non-trivial cycles to three. Then, the neighbor is obtained joining the solution obtained by solving the formulation $ILP^{(3)}$ over the set of isolated vertices with the previous fixed non-trivial cycles.

Instead of analyzing all the possible neighbors obtained by removing two non-trivial cycles, we decided to analyze NN neighbors for a given solution. We set NN as two times the amount of non-trivial cycles. The reason is to limit to a linear exploration of the search space related to the number of non-trivial cycles and not to a quadratic one, as it would be if exploring all the neighbors obtained by removing each two non-trivial cycles.

Moreover, we explored this neighborhood using the first improvement strategy that stops the local search as soon as the first neighbor that improves the current solution is found.

Algorithm 3: *Local_Search*

```
Input: A rainbow cycle cover $R$.
Output: A rainbow cycle cover $R^*$.
$nN \leftarrow 0$
$R^* \leftarrow R$
while $nN < NN$ do
    $nN \leftarrow nN + 1$
    $N \leftarrow$ non-trivial cycles in $R^*$
    $T \leftarrow$ trivial cycles in $R^*$
    $C, C' \leftarrow$ any two cycles in $N$
    $N \leftarrow N \setminus \{C \cup C'\}$
    $T' \leftarrow$ all vertices in the cycles $C$ and $C'$
    $H \leftarrow$ induced graph by $T \cup T'$
    $R' \leftarrow ILP^{(3)}(H)$
    if $|N| + |R'| < |R^*|$ then
        $R^* \leftarrow N \cup R'$
        $nN \leftarrow 0$
return $R^*$
```

3.3 Perturbation

The goal of the perturbation in the ILS metaheuristics is to escape from local optima and to introduce diversity in the search space. To achieve this, $\alpha|N|$ ($\alpha \in (0, 1]$) non-trivial cycles of the set of non-trivial cycles of the solution are removed. Then, the solution is reconstructed in a similar way to the one performed in Algorithm 2. The main difference is that when the $ILP^{(2)}$ procedure is executed the first time, restrictions are added to avoid forming the removed cycles.

```
Algorithm 4: Perturbation
Input: A rainbow cycle cover $R$.
Output: A perturbed rainbow cycle cover $R^*$.
$N \leftarrow$ non-trivial cycles in $R$
$LC \leftarrow$ list formed by $\alpha|N|$ cycles in $N$
$LR \leftarrow$ list of restrictions to avoid forming cycles in $LC$.
$N \leftarrow N \setminus LC$
$T \leftarrow$ trivial cycles in $R$ and the vertices in $LC$
$R \leftarrow \emptyset$
$flag \leftarrow 0$
while $N \neq \emptyset$ do
    $R \leftarrow R \cup N$
    $H \leftarrow$ induced graph by $T$
    if $flag = 0$ then
        $R' \leftarrow ILP^{(2)}(H, LR)$
        $flag \leftarrow 1$
    else
        $R' \leftarrow ILP^{(2)}(H)$
    $N \leftarrow$ non-trivial cycles in $R'$
    $T \leftarrow$ trivial cycles in $R'$
$R \leftarrow R \cup T$
return $R$
```

4 Experimental analysis

In this section, the results obtained by the proposed matheuristic are presented. We analyze the same set of instances used in [12] and some of the instances used in [13]. These instances correspond to graphs with number of vertices ranging from 20 to 100, and densities varying between 0.1, 0.2 and 0.3. The number of colors is always smaller than the number of vertices and varies between 3 and 18. The tests were developed on an Intel (R) Core i5-4460S CPU @ 2.90GHz, with 6 Mb cache and 8 Gb of RAM using the operating system Fedora 22 and all methods were programmed in C ++ language using the gcc compiler.

The IBM ILOG CPLEX 12.6 was used to solve the problem using the ILP formulation with a single thread of execution and 10800 seconds as time limit. All other CPLEX parameters were left to their default values.

As the graphs of these instances have small dimensions, we decided to use $\alpha = \frac{1}{3}$ in the perturbation procedure. The IBM ILOG CPLEX 12.6 was used to solve the model $ILP^{(q)}$with a single thread of execution and 30 seconds as time limit for graphs with 50 vertices or less and 60 seconds for graphs with more than 50 vertices.

4.1 Summary of computational results

The computational results are summarized in Table 1, where each line presents average statistics over five instances. The first four columns report the group identification (**ID**), number of vertices (**n**), number of edges (**m**) and number of colors (**l**) respectively. Columns 5 and 6 present, respectively, the average number of rainbow cycles found and average execution time in seconds obtained

Table 1: Experiments results for $ILS^{(ILP)}$.

ID	n	m	l	ILP	ILP-time	$ILS^{(ILP)}$	$ILS^{(ILP)}$-time	gap
1	20	39	3	18.0	0.00	18.0	0.12	0.0
2			6	13.6	0.01	13.6	1.19	0.0
3			11	10.6	0.03	10.6	1.66	0.0
4	20	58	3	14.8	0.14	14.8	1.25	0.0
5			6	9.8	0.23	9.8	4.37	0.0
6			12	6.8	0.24	6.8	8.85	0.0
7	20	77	4	10.0	0.41	10.0	1.79	0.0
8			7	6.6	0.53	6.6	6.81	0.0
9			13	4.6	0.36	4.6	9.30	0.0
10	30	74	4	21.8	0.31	21.8	1.44	0.0
11			7	19.4	0.45	19.4	9.87	0.0
12			13	15.4	0.32	15.4	9.83	0.0
13	30	117	4	18.2	4.43	18.2	2.66	0.0
14			7	13.0	4.09	13.0	8.36	0.0
15			14	9.6	2.57	9.6	29.02	0.0
16	30	161	4	15.2	94.43	15.4	3.94	0.2
17			8	8.0	64.15	8.4	13.96	0.4
18			15	5.4	8.67	5.4	55.04	0.0
19	40	118	4	30.6	1.31	30.6	2.79	0.0
20			7	24.8	2.77	24.8	9.32	0.0
21			14	20.4	1.53	20.4	12.58	0.0
22	40	196	4	25.0	83.04	25.2	5.09	0.2
23			8	15.8	1717.42	15.8	15.41	0.0
24			16	11.0	23.78	11.0	41.63	0.0
25	40	274	5	14.6	1322.37	15.0	7.44	0.4
26			9	8.4	2388.52	9.4	22.59	1.0
27			17	5.2	561.52	5.6	55.63	0.4
28	50	173	4	35.8	35.80	35.8	3.15	0.0
29			8	30.2	60.29	30.2	22.59	0.0
30			15	23.0	45.75	23.0	33.53	0.0
31	50	295	5	25.2	825.95	26.0	8.72	0.8
32			9	18.2	505.61	19.0	24.32	0.8
33			17	12.4	938.91	12.6	50.65	0.2
34	50	418	5	19.6	7503.57	20.4	11.42	0.8
35			9	12.2	10306.44	13.4	25.96	1.2
36			18	6.4	2931.14	6.6	91.83	0.2
37	100	595	5	69.4	...	**69.0**	32.85	-0.4
38			10	56.4	...	**54.0**	151.77	-2.4
39			19	46.2	...	**46.0**	267.82	-0.2
40	100	1090	6	46.4	...	**43.8**	116.26	-2.6
41			11	36.6	...	**32.6**	335.23	-4.0
42			21	34.0	...	**23.4**	312.22	-10.6
43	100	1585	6	44.4	...	**33.6**	92.47	-10.8
44			11	33.6	...	**20.8**	445.28	-12.8
45			21	30.0	...	**10.8**	463.62	-19.2

by solving the problem using the mathematical formulation ILP presented in Section 2. Columns 7 and 8 present the same information obtained using the matheuristic $ILS^{(ILP)}$. The last column shows the gap, calculated as:

$$gap = \frac{ILP - ILS^{(ILP)}}{ILP}$$

When the matheuristic $ILS^{(ILP)}$ obtains a better result than the exact method ILP, values are highlighted in boldface.

For graphs with 20 and 30 vertices, the branch-and-cut used few seconds to obtain the exact solution. In these graphs, the matheuristic $ILS^{(ILP)}$ has spent more time and obtained the optimal values for all groups, except for the groups with id = 16 and id = 17. For graphs with 40 and 50 vertices, the matheuristic presents a very small gap related to the exact method and a maximum average time of 91.83 seconds (reached in the group with id = 36). The proposed method is quite effective in groups of graphs with 100 vertices. In these groups, the branch-and-cut fails to find the optimal values for all instances in a time limit of 3 hours (10800 seconds) of execution, and the proposed matheuristic got much better results in a maximum average time of 463.62 seconds.

5 Conclusions

In this work, a based ILS matheuristic was proposed for the Rainbow Cycle Cover problem. The developed method uses the restricted model to execute the different stages. According to the results, the matheuristic obtains very precise values for instances up to 50 vertices. Moreover, for graphs with 100 vertices, the performance of the matheuristic was superior to that of the branch-and-cut in terms of time and quality of the results.

References

1. Hassin, R., Monnot, J., Segev, D.: Approximation algorithms and hardness results for labeled connectivity problems. Journal of Combinatorial Optimization **14** (2007) 437–453
2. Kano, M., Li, X.: Monochromatic and heterochromatic subgraphs in edge-colored graphs-a survey. Graphs and Combinatorics **24** (2008) 237–263
3. Krumke, S.O., Wirth, H.C.: On the minimum label spanning tree problem. Information Processing Letters **66** (1998) 81–85
4. Lai, X., Zhou, Y., He, J., Zhang, J.: Performance analysis of evolutionary algorithms for the minimum label spanning tree problem. IEEE Transactions on Evolutionary Computation **18** (2014) 860–872
5. Alexeev, B.: On lengths of rainbow cycles. Journal of Combinatorics **13** (2006) R105
6. Li, X., Zhang, X.: On the minimum monochromatic or multicolored subgraph partition problems. Theoretical Computer Science **385** (2007) 1–10
7. Zhang, X., Zhang, Z.B., Broersma, H., Wen, X.: On the complexity of edge-colored subgraph partitioning problems in network optimization. Discrete Mathematics and Theoretical Computer Science **17** (2016) 227
8. Moreno, J., Frota, Y., Martins, S.: A note on the rainbow cycle cover problem. Manuscript submitted for publication (2018)
9. Maniezzo, V., Stützle, T., Voß, S.: Matheuristics, volume 10 of annals of information systems (2010)
10. Martın, B., Sánchez, Á., Beltran-Royo, C., Duarte, A.: A matheuristic approach for solving the edge-disjoint paths problem. Matheuristics 2016 (2016) 25
11. Dupin, N., Talbi, E.G.: Matheuristics for the discrete unit commitment problem with min-stop ramping constraints. Matheuristics 2016 (2016) 72
12. Silvestri, S., Laporte, G., Cerulli, R.: The rainbow cycle cover problem. Networks **68** (2016) 260–270
13. Carrabs, F., Cerrone, C., Cerulli, R., Silvestri, S.: The rainbow spanning forest problem. Soft Computing (2017) 1–12
14. Lourenço, H.R., Martin, O.C., Stützle, T.: Iterated local search: Framework and applications. In: Handbook of Metaheuristics. Springer (2010) 363–397

Solving Multiple Sequence Alignment Problem Using a Discrete Hybrid Particle Swarm Optimization Algorithm

H. Deschênes[1] and C. Gagné[1]

1. Département d'informatique et de mathématiques, Université du Québec à Chicoutimi
555 boulevard de l'Université, Saguenay (QC), Canada, G7H 2B1
{hugo.deschenes, caroline.gagne}@uqac.ca

Keywords: comprehensive learning PSO, cooperative learning PSO, global alignment, discrete optimization, hybridization, metaheuristic, discretization.

1 Introduction

Metaheuristics are widely used methods for solving complex optimization problems of continuous or combinatorial nature. Some population-based methods conceived specifically for solving continuous optimization problems have been adapted and discretized [1]. They have therefore the ability to solve a wider variety of problems (including binary and discrete). The Particle Swarm Optimization algorithm (PSO) is one of those continuous methods that have been adapted over the years by the use of discretization techniques to solve problems with integer variables [1].

The Multiple Sequence Alignment (MSA) is an NP-hard [2] optimization problem of bioinformatics. The goal is to get an alignment of biological sequences which gets the highest resemblance score. This is done by adding or removing gaps in the chosen sequences and by evaluating the alignment according to the fitness function using the PAM250 or BLOSUM62 matrix. The best location of these gaps needs to be found in order to get the best result. It has an increasing difficulty the more sequences are used and the larger they are. The PSO is one of the methods that have been found to be quite efficient in solving MSA related problems [3]. Even if this metaheuristic has been enhanced over the years, it has however demonstrated premature convergence and difficulties in solving high dimensional problems [3, 4]. The goal of this paper is to help enhance the results provided by PSO algorithms on high-dimensional discrete problems like the MSA. A discrete hybrid PSO method based on the Comprehensive Learning PSO (CLPSO) [5] and the Cooperative Learning PSO (CoLPSO) [4] is proposed in this paper and will be compared to other PSO methods used to solve the MSA.

2 Related Works on the PSO

The PSO [6] is an evolutionary algorithm where a population improves with the help of all individuals on each generation. It replicates the behavior of birds in a flock or fish in a school as they are looking for food. It uses cooperation inside the swarm (flock, school) so that every particle (bird, fish) can improve with the aid of the rest of the swarm. When a particle is looking for a good solution (food source), it exchanges information with the swarm on the solutions observed so that all particles can potentially help each other to reach an optimum. A particle i in generation t evolves in a swarm by updating its velocity v_i^t using Equation 1, where numbers r_1 and r_2 are randomly generated between 0 and 1. A particle knows its best position visited so far p_i and the best global position of the swarm g. All components of a particle are pondered according to its inertia weight w and two coefficients, c_1 and c_2. The particle is then updated using its current position x_i^t and the updated velocity calculated with Equation 2.

$$v_i^{t+1} = w \cdot v_i^t + c_1 \cdot r_1 \cdot \left(p_i - x_i^t\right) + c_2 \cdot r_2 \cdot \left(g - x_i^t\right) \tag{1}$$

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{2}$$

The PSO algorithm evolved through the years. Variants of this metaheuristic have been proposed to improve the results obtained on different sets of problems. The CLPSO is a PSO variant which is efficient in getting the particles out of local optima when optimizing continuous multimodal functions [5]. It adds a comprehensive learning process at the beginning of the algorithm which is triggered when solutions stop improving for a fixed number of generations. It also modifies Equation 1 from the PSO by removing the global component of the formulation and replacing it with Equation 3. The CoLPSO is a PSO variant developed to help optimize high scale continuous optimization problems [4]. Considering that the performance of the PSO declines the more dimension is added to the problem, this variant gets around this weakness by separately optimizing each dimension of the standard PSO algorithm. The loop iterating the particles and the one iterating the dimensions are swapped during implementation. It is conceived so that the swarm is comprised of as many sub-swarms as there are dimensions in the problem. Each of these sub-swarms act on its own and is made of many one-dimensional particles. Communications are made between swarms once every generation to communicate best solutions so far and to build the global solution.

$$v_i^{t+1} = w \cdot v_i^t + c_1 \cdot r_1 \cdot \left(p_i - x_i^t\right) \tag{3}$$

Maitra and Chatterjee [7] proposed an hybrid named HCoCLPSO made from the CLPSO and the CoLPSO. This hybrid combines the comprehensive learning process from the CLPSO and the sub-swarms system made of one-dimensional particles from the CoLPSO. The HCoCLPSO has been applied on the image segmentation problem using multilevel thresholding and gets good results on high-dimensional continuous optimization problems. This paper uses a discretized version of this hybrid to solve the MSA.

3 Proposed methodology

Some PSO methods have been used to solve the MSA. In most cases, an efficient way to do this is to fill the particles with gaps positions from the sequences that are needed to align them. Considering that a solution to common alignment problems contains from 20% [8] to 40% [9] gaps, it is possible to deduce the maximum number of gaps that can be added to an entire alignment. This number determines the dimension D of a particle. The codification of a solution used in this paper is similar to Moustafa and *al.* [3]. In their work, a position is represented by a single vector with the coordinates of every gap in an alignment. An additional component is added to each particle indicating how many gaps are needed in each sequence to fill the maximum length of the alignment. In this paper, the maximum length includes the extended 20%. The goal of this PSO in solving the MSA is to move the gaps to different locations in the alignment in order to find the highest score from the Sum of Pair (SOP) function. The SOP is used to increase the similarity match between every two sequences in an alignment. For an aligned dataset of m sequences, the score is calculated for every two sequences S_a and S_b in a position x_i^t of a particle i as shown in Equation 4. In this work, the pairwise alignments are evaluated according to the scoring matrix PAM250. If two characters are similar, a positive score is added to the total fitness of the alignment. If there is a mismatch or a character is aligned with a gap, a penalty is subtracted as a negative score.

$$Pairwise\ alignment\ score = \sum_{d=1}^{D} score\left(S_a\left(x_{i,d}^t\right), S_b\left(x_{i,d}^t\right)\right) \tag{4}$$

We first discretize the HCoCLPSO algorithm with two different techniques. These two techniques have been chosen for their eligibility to be used to solve the MSA and their good performances. The first technique keeps the PSO formulation for continuous optimization and adds a transformation process to a particle's position before it is evaluated by the fitness function [10]. These processes are called *Forward transformation* (Equation 5) and *Backward transformation* (Equation 6). They allow a position to convert its real-size values to an integer format and vice-versa. Each converted value represents the location of a gap in a sequence. The factor h in these equations has been established to 100 by Cuevas and *al.*

$$x_i' = -1 + \frac{x_i h 5}{10^3 - 1} \tag{5}$$

$$int[x_i'] = \frac{(1 + x_i')(10^3 - 1)}{5h} \tag{6}$$

The second technique is the one used by Moustafa and *al.* [3]. It defines each number contained in a position of the PSO as integers representing the locations of all gaps in an alignment. The velocity is still defined as real values as they represent a displacement vector applied to the position of a particle. The values contained in the new velocity v_i^{t+1} calculated in Equation 1 are then rounded before being added to a particle's position in Equation 2. In both techniques, the fitness function takes in consideration the possibility that gaps get aligned together, thus producing an alignment with an empty column that must not penalize the global alignment score.

The proposed methods is tested on some *Balibase* benchmarks of different lengths [11]. The score obtained with SOP is compared with two other methods from the literature: FTLPSO [3] and TLPSO-MSA [12]. Both of these methods compare themselves with well-known MSA tools such as CLUSTAL Omega, CLUSTAL W2, TCOFFEE, KALIGN and DIALIGN-PFAM.

4 Conclusion

In this extended abstract, the main concepts of solving the MSA with PSO have been highlighted. Two PSO variants are presented (CLPSO and CoLPSO) along with a hybrid version made of these two. Considering that the MSA is a discrete problem, two ways to discretize the PSO are presented. The experimental process is approached in order to compare the experimental results with methods from the literature (FTLPSO and TLPSO-MSA). The purpose of this contribution is to enhance the application of PSO and other population-based methods on a wider variety of problems by using variants and discretization techniques. It also aims at improving the PSO performances on discrete and high-scale optimization problems such as the MSA.

5 References

[1] J. Krause, J. Cordeiro, R.S. Parpinelli, H.S. Lopes, A survey of swarm algorithms applied to discrete optimization problems, Publisher, City, 2013.

[2] L. Wang, T. Jiang, On the complexity of multiple sequence alignment, Publisher, City, 1994.

[3] N. Moustafa, M. Elhosseini, T.H. Taha, M. Salem, Fragmented protein sequence alignment using two-layer particle swarm optimization (FTLPSO), Publisher, City, 2017.

[4] F. Van den Bergh, A.P. Engelbrecht, A cooperative approach to particle swarm optimization, Publisher, City, 2004.

[5] J.J. Liang, A.K. Qin, P.N. Suganthan, S. Baskar, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, Publisher, City, 2006.

[6] Y. Shi, R. Eberhart, A modified particle swarm optimizer, in: Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on, 1998, pp. 69-73.

[7] M. Maitra, A. Chatterjee, A hybrid cooperative–comprehensive learning based PSO algorithm for image segmentation using multilevel thresholding, Publisher, City, 2008.

[8] H.X. Long, W.B. Xu, J. Sun, W.J. Ji, Multiple Sequence Alignment Based on a Binary Particle Swarm Optimization Algorithm, in: 2009 Fifth International Conference on Natural Computation, 2009, pp. 265-269.

[9] F. Xu, Y. Chen, A Method for Multiple Sequence Alignment Based on Particle Swarm Optimization, in: D.-S. Huang, K.-H. Jo, H.-H. Lee, H.-J. Kang, V. Bevilacqua (Eds.) Emerging Intelligent Computing Technology and Applications. With Aspects of Artificial Intelligence: 5th International Conference on Intelligent Computing, ICIC 2009 Ulsan, South Korea, September 16-19, 2009 Proceedings, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 965-973.

[10] E. Cuevas, D. Zaldivar, M. Pérez-Cisneros, M. Ramírez-Ortegón, Circle detection using discrete differential evolution optimization, Publisher, City, 2011.

[11] A. Bahr, J.D. Thompson, J.C. Thierry, O. Poch, BAliBASE (Benchmark Alignment dataBASE): enhancements for repeats, transmembrane sequences and circular permutations, Publisher, City, 2001.

[12] S. Lalwani, R. Kumar, N. Gupta, A novel two-level particle swarm optimization approach for efficient multiple sequence alignment, Publisher, City, 2015.

Nature-inspired Deployment for Context-aware Services

Ichiro Satoh

National Institute of Informatics
2-1-2 Hitotsubashi Chiyoda-ku Tokyo, 101-8430, JAPAN
ichiro@nii.ac.jp

Abstract. A nature-inspired context-aware service platform in IoT or ambient computing environments is presented. Context-aware services are required to adapt themselves to environmental changes. We have developed adaptation mechanisms that utilize gravity and repulsive forces between users and services. Our system is constructed as a non-centralized distributed service platform for executing context-aware services and managing location sensing systems. It can dynamically deploy software for defining services at computers in accordance with contexts, e.g., the locations of users in the real world. We evaluated the system using real applications with real users in a museum.

1 Introduction

Our final objective is to construct a general-purpose infrastructure for providing numerous users with context-aware services in large-scale public spaces, e.g., city-wide spaces. For example, city-wide computing environments consist of numerous computing devices and provide a variety of services to diverse users, some of them may behave unexpectedly. THe infrastructure also need to be scalable and robust, even when it consists of numerous devices that may occasionally be inactive. Like large-scale distributed systems, the scale and complexity of such a context-aware system is beyond our ability to manage using conventional approaches such as centralized or top-down approaches. Nature-inspired approaches enable us to manage large-scale context-aware service infrastructures in a non-centralized and peer-to-peer manner.

This paper introduces a nature-inspired approach for providing context-aware services. The approach is built on two metaphors from nature: *gravitational* and *repulsive* forces between software for defining services and target entities, including people or spaces that the services are provided for in the real world (Fig. 1). The former deploys software for defining services at computers nearby the targets and executing them there. It is used as a relocation between users and services. The latter prevents software for defining services from being at computers nearby the locations of the targets. It is used as a relocation technique between similar services. Some of the metaphors in the approach were discussed in our previous paper [8]. In this paper, we address an application of the metaphors to a context-aware system in the real world.

The approach is constructed as a general-purpose platform for context-aware services combining the two metaphors. To ensure independence from the underlying location systems, the platform introduces virtual counterparts for the target entities and spaces and the two metaphors represent the relationships between software for defining services and virtual counterparts corresponding to their targets. This paper presents the design and implementation of the platform and our evaluation of our nature-inspired approach for context-aware services through a visitor guide in a real museum as a case study.

In Section 2 of this paper briefly describes our nature-inspired approach and its requirements. Section 3 describes the design and implementation of the approach and Section 4 presents our initial experiences with the approach. Section 5 surveys related work. We conclude in Section 6 with a briefly summary and mention of future work.

2 Basic Approach

Although our final objective is to provide a context-aware service platform in the real world. In this paper we address the nature-inspired approach behind the platform.

2.1 Requirements

First we discuss the requirements of our application, which is a visitor guide system in a science museums in this paper. The goal of a science museum is to provide experiences to visitors that will enhance their

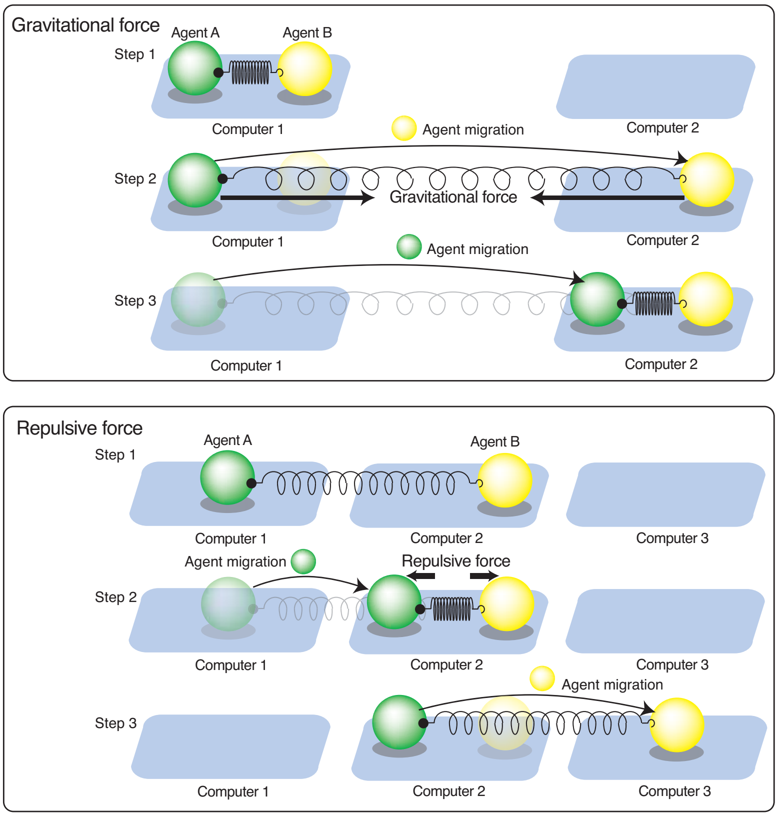


Fig. 1. Gravitational and repulsive policies

knowledge of science from the exhibitions, and not their experiences with ambient-computing, including context-awareness. Not all visitors have sufficient knowledge about the exhibits in museums should be provided, so annotations about the exhibits on par with their knowledge and interests should be provided. Since their knowledge and experience are varied, they may become puzzled (or bored) if the annotations provided are beyond (or beneath) their knowledge or interest. Supplementary annotation services dependent on the visitors are needed in order for them to fully understand the exhibits.

The platform itself should enable context-aware annotation services to be provided from both stationary and mobile terminals. However, our target museum wants to primarily use stationary terminals rather than mobile terminals because if services are provided from mobile terminals, visitors tend to pay attention to the terminals instead of the exhibits. Some visitors may not have their own mobile terminals, e.g., cellular phones and smartphones and they tend to dislike complex operations and interactions with systems. It is also difficult for some visitors, particularly children, the elderly, and handicapped people, to interact with annotation services.

Therefore, such services should be executed at nearby computers to minimize latency in communication between user-interface devices and server-side computers for reasons of response. Computers in public spaces may have only limited resources, such as restricted levels of CPU power and memory. They cannot support all the services that may be needed beforehand. Software for defining annotation services should be deployed at such computers only while those services are needed.

Museums are not expressly large-scale spaces and do not have as many users as say, entire cities. Nevertheless, our final goal is a city-wide context-aware service platform that will provide a variety of services to massive numbers of users from numerous heterogeneous computers. Therefore, our platform is designed for a city-wide context-aware scenario.

2.2 Nature-inspired context-aware service platform

The key idea behind the platform presented in this paper is to separate between application-specific services for users from context-aware policies of such services. There have been many attempts to provide context-aware services, but these services depends so heavily on their target contexts so that most of them cannot be used for other contexts. However, context-aware services themselves are common. For example, location-aware annotation services on exhibits designed for running on mobile terminals are often activated only when their users are close to the exhibits. At the same time, the contents of the annotation services themselves may be able to be used on stationary terminals. To reuse such services in other context, the platform provides contextual conditions that the services should be activated as policies defined outside the services. The policies can be classified into two types: *gravitational* and *repulsive* forces between services and the target entities and spaces.

Virtual counterparts To introduce the metaphors of gravitational and repulsive forces to context-aware services, we abstract away the underlying systems, including location-sensing systems. Our platform has the following two kinds of agents, discussed below.

- Physical entities, people, and spaces can have their digital representations, called *virtual-counterpart* agents, in the platform. Each virtual-counterpart is automatically deployed at computers close to its target entity or person or within the space. For example, a virtual-counterpart for users can store per-user preferences and record user behavior, e.g., exhibits that they have looked at.
- The platform assumes an application-specific service to be defined in a software component and executes the component as an autonomous entity, called a *service-provider* agent, individually for each agent. In the current implementation, existing Java-based software components, e.g., JavaBeans, can define our services.

The first and second agent are executed in runtime systems and can be dynamically deployed at the runtime systems different computers. They are executed as mobile agents [10] that can travel from computer to computer under their own control. When a user approaches closer to an exhibit, our system detects that user migration by using location-sensing systems and then instructs that user's counterpart agent to migrate to a computer close to the exhibit.

Context-aware deployment as forces between agents As mentioned previously, we introduce nature-inspired agent deployment policies based on two metaphors: *gravitational* and *repulsive* forces. Virtual-counterpart and service-provider agents are loosely coupled so that they can be dynamically linked to others. The current implementation has two built-in *gravitational* policies:

- An agent has a *follow* policy for another agent. When the latter migrates to a computer or a location, the former migrates to the latter's destination computer or to a computer nearby.
- An agent has a *shift* policy for another agent. When the latter migrates to a computer or a location, the former migrates to the latter's source computer or to a computer nearby.

Each service-provider agent can have at most one *gravitational* policy. Although the *gravitational* policy itself does not distinguish between virtual-counterpart and service-provider agents, in the above policies, we often assume the former to be a service-provider agent and the latter to be a virtual-counterpart agent. For example, when a visitor stands in front of an exhibit, the underling location-sensing system detects the location of the visitor and then the visitor's virtual counterpart agent is deployed at a computer close to the current location. When service-provider agents declare follow policies for the counterpart agent, they are deployed at the computer or nearby computers.

The current implementation has two built-in *repulsive* policies. Each service-provider agent can have zero or more repulsive policies in addition to the *shift* policy.

- An agent has an *exclusive* policy for another agent. When the former and latter are running on the same computer or nearby computers, the former migrates to another computer.
- An agent has a *suspend* policy for another agent. When the former and the latter are running on the same computer or nearby computers, the former is suspended until the latter moves to another computer.

Each service-provider agent can have at the most one *repulsive* policy. To avoid the redundancy of agents whose services are similar at the same computers, we use *repulsive* force between service-provider agents.

3 Design and Implementation

Our platform consists of three parts: (1) context information managers, (2) runtime systems, and (3) mobile agents with nature-inspired deployment policies, as shown in Fig. 2. The first provides a layer of indirection between the underlying locating-sensing systems and agents. It manages one or more sensing systems to monitor contexts in the real world and provides neighboring runtime systems with up-to-date contextual information of its target entities, people, and places. The second is constructed as a distributed systems consisting of multiple computers, including stationary terminals and users' mobile terminals, in addition to servers. Each runtime system runs on a computer in the real world and is responsible for executing and migrating virtual-counterpart and service-provider agents with nature-inspired deployment policies. It evaluates the deployment policies of agents and then deploys the agents at runtime systems. The third is virtual-counterpart or service-provider agents, where the former offers application-specific content, which is attached to physical entities, people, and places, and the latter can be defined as conventional Java-based software components, e.g., JavaBeans.

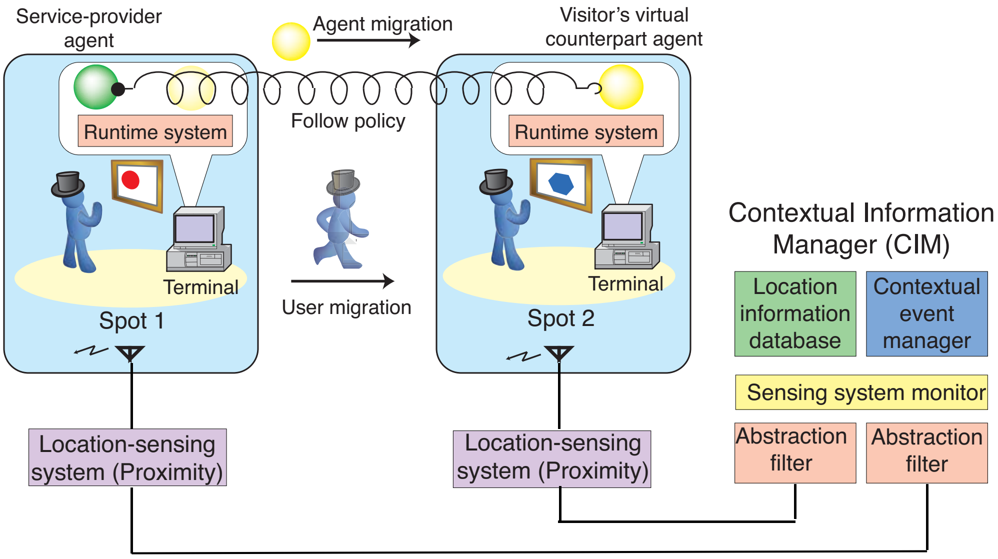


Fig. 2. Architecture.

3.1 Context information manager

Each context information manager (CIM) manages one or more sensing systems to monitor context in the real world, e.g., people and the locations of the target entities and people. For example, the current implementation of CIM supports several sensing systems, including active RFID-tag systems. Here we describe how the manager notifies runtime systems nearby the locations of people and physical entities through an active RFID-tag system.

CIM monitors the RFID-tag systems, detects the presence of tags attached to people and entities, and maintains up-to-date information on the identities of RFID tags that are within the zones of coverage of its RFID tag readers. To abstract away the differences between the underlying locating systems, each CIM maps low-level positional information from each of the locating systems into information in a symbolic model of the location. The current implementation represents an entity's location, called a *spot*, e.g., spaces of a few feet, which distinguishes one or more portions of a room or building. A CIM either polls its sensing systems or receives the events issued by the sensing systems or other CIMs. Each CIM has a database for mapping the identifiers of RFID tags and virtual counterparts corresponding to physical entities, people, and spaces attached to the tags. These database may maintain information on several tags. When a CIM detects the existence of a tag in a spot, it multicasts a message containing the identifier of the tag, the identifiers of virtual counterparts attached to the tag, and its own network address to nearby runtime systems. When there are multiple candidate destinations, each of the agents that is tied to a tag can select one destination on the basis of the profiles of the destinations. When the absence of a tag is detected in a spot, each CIM multicasts a message with the identifier of the tag and the identifier of the spot to all runtime systems in its current sub-network.

3.2 Runtime system

Runtime systems migrate agents to other runtime systems running on different computers through TCP channels using mobile-agent technology [10].

Agent execution and migration management Each runtime system is built on Java virtual machine (Java VM) version 1.7 or later, which conceals differences between the platform architectures of the source and destination computers (Fig. 3). It governs all the agents inside it and maintains the life-cycle state of each agent. When the life-cycle state of an agent changes, e.g., when it is created, terminates, or migrates to another runtime system, its current runtime system issues specific events to the agent.

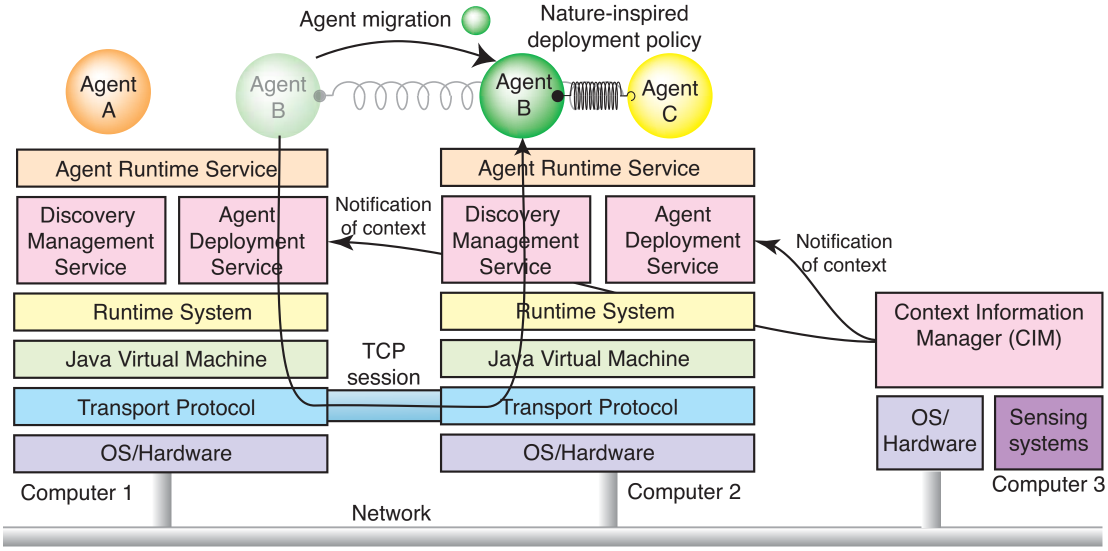


Fig. 3. Runtime system

When an agent is transferred over the network, not only its code but also its state is transformed into a bitstream by using Java's object serialization package and then the bit stream is transferred to the destination. Since the package does not permit the stack frames of threads to be captured, when an agent is deployed at another computer, its runtime system propagates certain events to to instruct it to stop its active threads. Arriving agents may explicitly have to acquire various resources, e.g., video and sound, or release previously acquired resources.

The system only maintains per-user profile information within those agents that are bound to the user. It instructs the agents to move to appropriate runtime systems near the user in response to his/her movements. Thus, the agents do not leak profile information on their users to third parties and they can interact with mobile users in a personalized form that has been adapted to respective, individual users. The runtime system can encrypt agents to be encrypted before migrating them over a network and then decrypt them after they arrive at their destinations.

Agent deployment management Our nature-inspired deployment policies are managed by runtime systems without a centralized management server. When a runtime system receives the identifiers of virtual counterparts corresponding to physical entities, people, and spaces attached to newly visiting tags, it discovers the locations of the virtual counterparts by exchanging query messages between nearby runtime systems.

Each runtime system periodically advertises its address to the others through UDP multicasting, and these runtime systems then return their addresses and capabilities to the runtime system through a TCP channel.[1] The procedure involves four steps. When an agent migrates to another agent's runtime system, each agent automatically registers its deployment policy with the destination. The destination sends a query message to the source of the visiting agent. There are two possible scenarios: the visiting agent has a policy for another agent or it is specified in another agent's policies. 3-a) Since the source in the first scenario knows the runtime system running the target agent specified in the visiting agent's policy , it asks the runtime system to send the destination information about itself and about neighboring runtime systems that it knows, e.g., network addresses and capabilities. If the target runtime system has retained the proxy of a target agent that has migrated to another location, it forwards the message to the destination of the agent via the proxy. 3-b) In the second scenario, the source multicasts a query message within current or neighboring sub-networks. If a runtime system has an agent whose policy specifies the visiting agent, it sends the destination information about itself and its neighboring runtime systems. 4) The destination next instructs the visiting agent or its clone to migrate to one of the candidate destinations recommended by the target, because this platform treats every agent as an autonomous entity.

3.3 Agent

Each mobile agent is attached to at most one visitor and maintains that visitors's preference information and programs to provide customized annotations. Each virtual counterpart agent keeps the identifier of the tag attached to its visitor.

Each agent in the current implementation is a collection of Java objects in the standard JAR file format and can migrate from computer to computer and duplicate itself by using mobile agent technology.[2] Each agent must be an instance of a subclass of the `Agent` class.

```
class Agent extends MobileAgent implements Serializable {
  void go(URL url) throws NoSuchHostException { ... }
  void duplicate() throws IllegalAccessException { ... }
  setPolicy(ComponnetProfile cref,
    MigrationPolicy mpolicy) { ... }
  setTTL(int lifespan) { ... }
  void setAgentProfile(AgentProfile cpf) { ... }
  boolean isConformableHost(HostProfile hfs) { ... }
  void send(URL url, AgentID id, Message msg)
    throws NoSuchHostException, NoSuchAgentException, ... { ... }
  Object call(URL url, AgentID id,
    Message msg) throws NoSuchHostException,
      NoSuchAgentException, ... { ... }
  ....
}
```

Each agent can execute `go(URL url)` to move to the destination specified as a `url` by its current platform, and `duplicate()` creates a copy of the agent, including its code and instance variables. The `setTTL()` specifies the life span, called the time-to-live (TTL), of the agent. The lifespan decrements TTL over time. When the TTL of an agent reaches zero, the agent automatically removes itself.

Our system enables agents to define the computational resources they require. When an agent migrates to the destination according to its policy, if the destination cannot satisfy the requirements of the agent, the platform system recommends candidates that are runtime systems in the same network domain to the agent. If an agent declares repulsive policies in addition to a gravitational policy, the platform system detects the candidates using the latter's policy and then recommends final candidates to the agent using the former policy, assuming that the agent is in each of the detected candidates.

[1] We assumed that the agents comprising an application would initially be deployed at runtime systems within a localized space smaller than the domain of a sub-network.

[2] JavaBeans can easily be translated into agents in this platform.

4 Experience

We concluded an experiment at the Museum of Nature and Human Activities in Hyogo, Japan using the proposed system.[3] As shown in Fig. 4 a sketch maps the various spots located in the museum. The experiment was carried out at four spots in front of specimens of taxidermied animals specifically i.e., a bear, deer, racoon dog, and wild boar. Each spot provided five different pieces of animation-based annotative content about the animals, e.g., its ethology, footprints, feeding, habitats, and features, and had a display and Spider's active RFID reader with a coverage range that roughly corresponded to the space, as shown in Fig. 5.

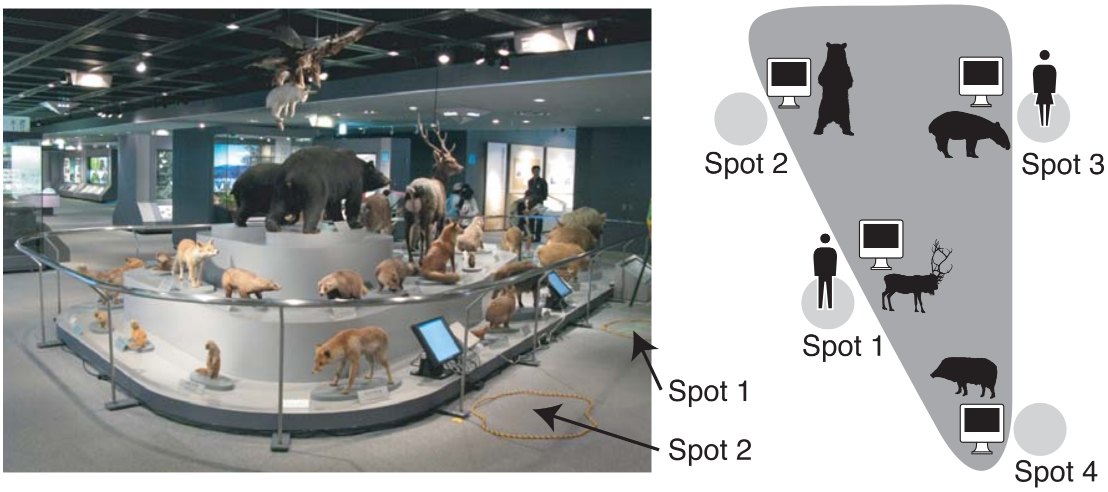


Fig. 4. Experiment at the Museum of Nature and Human Activities in Hyogo.

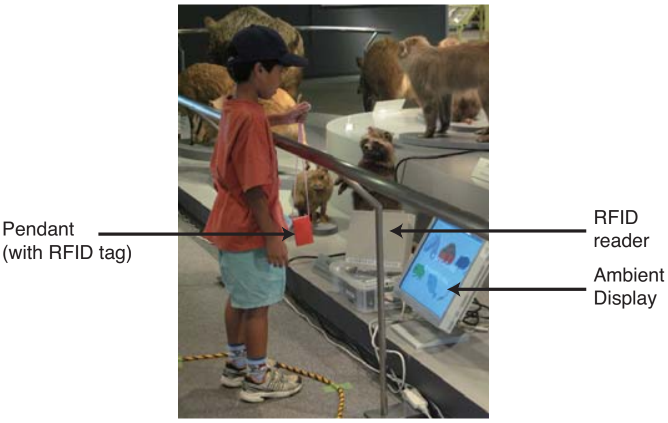


Fig. 5. Spot at Museum of the Nature and Human Activities in Hyogo.

All visitors in the experiment were provided with a pendant containing an RFID tag. The idea was designed to have visitors imagine that their agents, which were virtual owls, were inside their pendant. When visitors armed their pendants first started in the experiment, an operator input points of interest and

[3] The aim of this experiment is to provide context-aware services in a real space rather than evaluating this system. Consequently, not all the deployment policies are evaluated here.

the route by means of a Web browser running on a portable terminal. Visitors also created their own virtual counterparts and service-provider agents, where the latter agents had a *follow* deployment policy for the former agent. For example, suppose a visitor enters the spot with the specimen of a racoon dog. The visitors counterpart agent is deployed at a computer close to the specimen and then the service-provider agents are deployed at the computers according to their *follow* deployment policy.

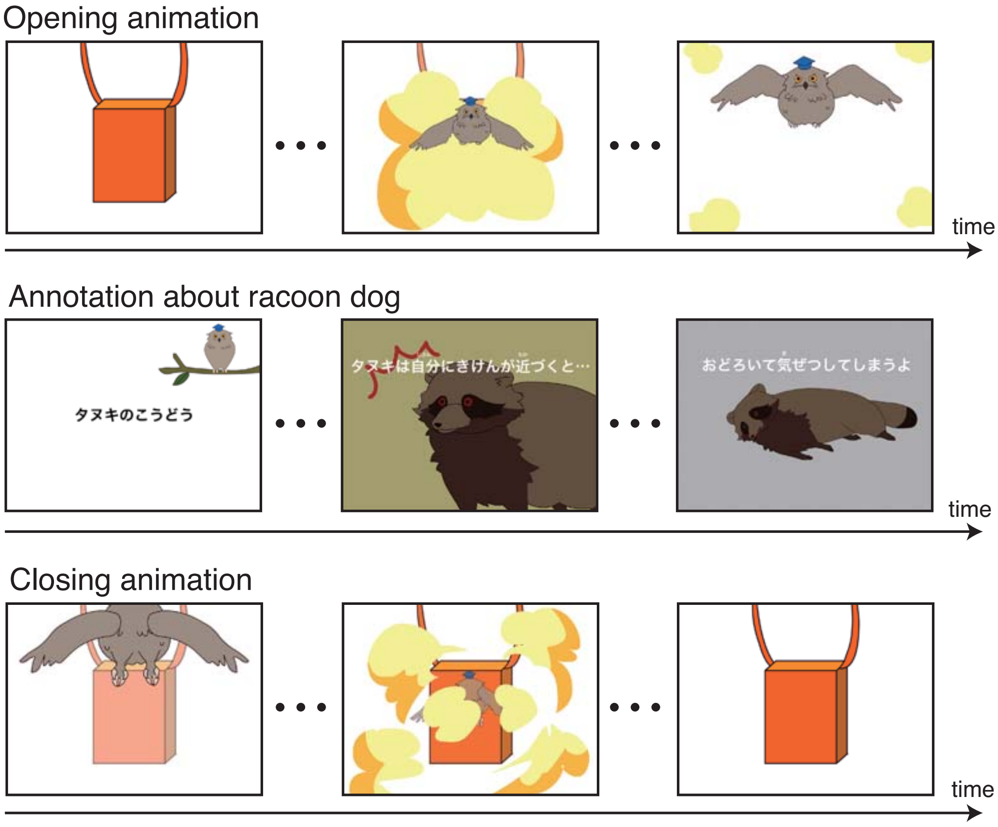

Opening animation
time
Annotation about racoon dog
タヌキのこうどう
タヌキは自分にきけんが近づくと…
おどろいて気ぜつしてしまうよ
time
Closing animation
time

Fig. 6. Animation of service-provider agent

As shown in Fig. 6, their service-provider agents play annotation about the ethology, footprints, feeding, habitats, and features of the animal (specimen) in their current spots. We simultaneously provided two different routes for visitors in order to evaluate the utility of our support for user navigation. Both routes guided visitors to various destination spots while ensuring that they walked around an exhibition booth consisting of four spots two or three times, as shown on the right in Fig. 4. That is, a visitor might visit the same spots two or three times depending on the navigation providing by the agents. Both the experiments offered visitors animation-based annotative content about the animal in front of them so that they could learn about it while observing the corresponding specimen.

In addition, we could dynamically change users' routes by replacing the current service-provider agents with other service-provider agents that declared deployment policies for the virtual counterpart agent corresponding to the users.

As there are many visitors in museums, we cannot adequately cope with conflicts caused by multiple users. For example, two visitors might simultaneously view and hear at most one annotation provided from a stationary computer in an exhibition space under the impression that the annotation is just for them. To solve this problem, we used *suspend* policy. First, visual representation of the agents, i.e., characters was used to help visitors to identify who the annotation is for, and second, each runtime system is equipped with a queuing mechanism for exclusively executing agents for multiple simultaneous users. When two users enter the same spot, the CIM sends two notification messages to the runtime system in that spot in the order in which they entered. The runtime system can send events to the agents bound to the two users in that order, or it can explicitly send an event to one of the agents. After the first has handled the event, it sends the same event to the second one. The current implementation supports several queuing policies, e.g., LIFO and synchronization among more than two users.

We conducted the experiment over a two-week period. Each day, more than 60 individuals or groups took part in the experiment. The majority of the participants were groups of families or friends aged from 7 to 16. Most visitors answered questionnaires consisting quizzes and feedback on the system along with information on their gender and age. Almost all the participants (more than 95 percent) provided positive feedback on the system. Typical feedback included statements like "We were very interested in or enjoyed the system", "We could easily answer the quizzes by moving between the spots", and "We gained detailed knowledge about the animals by watching them in front of where we were standing." Most visitors only paid attention to the colors of the agents, rather than the characters or visual effects. They appeared to be more interested in looking at the animal specimens than the agents.

To evaluate the performance overhead of the deployment policies presented in this paper, we implemented and evaluated a non-deployment policy version of the system. When this version detected the presence of a user at one of the spots, it directly deployed a service-provider agent instead of virtual counterpart agents. We measured the cost of migrating a null agent (a 5-KB agent, zip-compressed) and an annotation agent (1.2-MB agent, zip-compressed) from a source computer to a recommended destination computer that was recommended. The latency of discovering and instructing a virtual counterpart or service-provider agent attached to a tag after the CIM had detected the presence of the tag was 420 ms. Without any deployment policies, the respective cost of migrating the null and annotation agents between two runtime systems running on different computers over a TCP connection was 41 ms and 490 ms after instructing agents to migrate to the destination. When the null or annotation agent had a *follow* policy for the virtual counterpart agent, the respective cost of migrating the null and annotation agents between two runtime systems running on different computers over a TCP connection was 185 ms and 660 ms. These results demonstrate that the overhead of our deployment policy can be negligible in context-aware services.

Our experiment at the museum is a case study in our development of ambient-computing services in large-scale public spaces. However, we could not evaluate the scalability of the system in the museum because it consisted of only four terminals. Even so, we have a positive impression on the availability of the system for large-scale public services. This is because the experimental system could be operated without any centralized management system. The number of agents running or waiting on a single computer was bound to the number of users in front of the computer.

5 Related Work

This section discusses several bio-inspired approaches to distributed and multi-agents systems. A few attempts have provided infrastructures for real distributed systems, like ours. The Anthill project [1] by the University of Bologna developed a bio-inspired middleware for peer-to-peer systems composed of a collection of interconnected nests. Autonomous agents (called ants) can travel across the network trying to satisfy user requests same as ours. The project provided bio-inspired frameworks called Messor [5], and Bison [6]. Messor [5] is a load-balancing application of Anthill and Bison is a conceptual bio-inspired platform based on Anthill. The main difference between Anthill including its applications and our platform is that it introduces agents as independent entities which ours permits agents to be self-organized. The Co-Field project [4] by the University di Modena e Reggio Emilia proposed the notion of a computational force-field model for coordinating the movements of a group of agents comprising mobile devices, mobile robots, and sensors. However, the model only seems to be usable within the limits of simulation and not in a real distributed system. Our deployment policies are similar to the dynamic layout of distributed applications in the FarGo system [2], but FarGo's policies aim at allowing an agent to control others, whereas our policies aim at allowing an agent to describe its own individual migration, since our platform always treats agents as autonomous entities that travel from computer to computer under their own control. FarGo's policies may conflict when two agents can declare different relocation policies for a single agent. In contrast, our platform is free of any conflict because each agent can only declare a policy to relocate itself.

That system presented in this paper is an application of our previous bio-inspired system [8]. The system was a general-purpose test-bed platform for implementing and evaluating bio-inspired approaches over real distributed systems. It enabled each software agent to be dynamically organized with other agents and deployed at computers according to its own organization and deployment policies. In contrast, this paper addressed a practical system with nature-based approaches used in the real world with real users for real applications. We presented an outline of mobile agent-based services in public museums in our earlier versions of this papers [7,9], but did not describe any nature-inspired deployment policies in those works.

6 Conclusion

This paper presented a context-aware service platform with a nature-inspired or self-organizing approach. The system enabled two individual agent to specify one of the deployment policies as relocations between the agent and another. It can not only move individual agents but also a federation of agents over a distributed system in a self-organized manner. We evaluated the system by applying it to visitor-assistant services in a museum. When visitors move from exhibit to exhibit, the visitors' virtual counterpart agents can be dynamically deployed at computers close to the current exhibits to accompany the visitors via their virtual counterpart agents and play annotations about the exhibits. Visitors and service-provider agents are loosely coupled because the agents are dynamically linked to the virtual counterpart agents corresponding to them by using our deployment policies.

There are still a couple of issues that need to be resolved. For example, some readers may consider that the non-centralized management architecture we used was not needed to operate the context-aware visitor-guide services in a museum. However, our final goal is to provide large-scale context-aware services with nature-inspired centralized management approaches in large spaces, e.g., cities. We therefore need to demonstrate the scalability of the platform for large-scale context-aware services.

References

1. O. Babaoglu and H. Meling and A. Montresor, Anthill: A Framework for the Development of Agent-Based Peer-to-Peer Systems, Proceeding of 22th IEEE International Conference on Distributed Computing Systems, July 2002.
2. O. Holder, I. Ben-Shaul, and H. Gazit, System Support for Dynamic Layout of Distributed Applications, Proceedings of International Conference on Distributed Computing Systems (ICDCS'99), pp 403-411, IEEE Computer Soceity, 1999.
3. B. Horling, and V. Lesser, and R. Vincent, Multi-Agent System Simulation Framework Proceeding of IMACS World Congress 2000 on Scientific Computation, Applied Mathematics and Simulation, August 2000.
4. M. Mamei, L. Leonardi, F. Zambonelli, Co-Fields: A Unifying Approach to Swarm Intelligence, International Workshop on Engineering Societies in the Agents World (ESAW 2002), Lecture Notes in Computer Science, vol. 2577, Springer Verlag 2003.
5. A. Montresor, H. Meling, and O. Babaoglu, Messor: Load-Balancing through a Swarm of Autonomous Agents, Proceedings of International Workshop on Agents and Peer-to-Peer Computing, July 2002.
6. A. Montresor and O. Babaoglu, Biology-Inspired Approaches to Peer-to-Peer Computing in BISON Proceedings of International Conference on Intelligent System Design and Applications, August 2003.
7. I. Satoh, Context-aware Agents to Guide Visitors in Museums, in Proceedings of 8th International Conference Intelligent Virtual Agents (IVA'08), Lecture Notes in Artificial Intelligence (LNAI), vol.5208, pp.441-455, September 2008.
8. I. Satoh, Test-bed Platform for Bio-inspired Distributed Systems, in Proceesings of 3rd International Conference on Bio-Inspired Models of Network, Information, and Computing Systems, November 2008.
9. I. Satoh, A Context-aware Service Framework for Large-Scale Ambient Computing Environments, in Proceedings of ACM International Conference on Pervasive Services (ICPS'09), pp.199-208, ACM Press, July 2009.
10. I. Satoh: "Mobile Agents," Handbook of Ambient Intelligence and Smart Environments, pp.771-791, Springer 2010.

New hybrid differential evolution algorithm for multiple objective optimization

C. Gagné[1], M. Fournier[1], M. Gravel[1] and A. Sioud[1]

1. Université du Québec à Chicoutimi (UQAC)
555 boul. Université, Saguenay (QC), Canada, G7H 2B1
{caroline.gagne, mathieu.fournier1, marc.gravel, aymen.sioud}@uqac.ca

Keywords : metaheuristics, differential evolution, invasive weed optimization, hybridization, continuous problems.

1 Introduction

Many real world optimization problems, both in combinatorial and continuous optimization, involve simultaneous optimization of multiple incommensurable and often competing objective functions. For these optimization problems, there are usually many solutions of interest, each of which represents some compromise among objective functions, rather than a single optimum. These solutions, widely known as Pareto front, are optimal in the broader sense that no other solution in the search space is better over all of the objectives. Consequently, we can define two goals in multiple-objective (MO) optimization: (i) discovering solutions as close to the Pareto front as possible; and (ii) finding solutions as diverse as possible in the obtained solutions set. Satisfying these two goals is a challenge for any MO algorithm [1]. In recent years, the adaptation of metaheuristics has been proven to be a winning alternative for the simultaneous optimization of several objectives. Among others, the extensions of the genetic algorithm [2] and the differential evolution (DE) [3] inspired respectively the algorithms NSGAII [1] and DEMO [4].

Among MO problems in industry, those with continuous variables require a particular exploration of the search space in order to approach as close as possible the optimal real values. The DE algorithm, which is recognized for having a global search capability [5], proposes to consider solutions as vectors and to use vector displacement to explore solutions. Other population-based metaheuristics offer more exploitation capability, such as the invasive weed optimization (IWO) algorithm [6]. This metaheuristic is inspired by the proliferation of seeds in an environment. The analogy with nature makes it possible to translate *r*-selection and *K*-selection in order to make exploration at the beginning of resolution and then to make more exploitation. State of the art [7, 8] are available on a variety of metaheuristics adapted to solve continuous problems.

The main contribution of this paper is introducing a new powerful hybrid algorithm involving DE to solve continuous MO problems. This hybridization is carried out with the IWO algorithm in order to balance the exploration and exploitation capabilities in the solution search process. This new algorithm, noted as IWODEMO, aims at the two goals of MO metaheuristic, namely to converge towards the Pareto-optimal front and to find diversified solutions. To the best of our knowledge, IWO multi-objective algorithms currently do not offer a comparison with DE methods [9, 10]. Another part of the contribution is then to propose a comparative study under equitable experimental conditions. To do this, the proposed algorithm is compared to three algorithms from the literature (NSDE, DEMO and ADE-MOIA) to solve the same group of instances, for an equivalent computing power and with the maximum of common components.

2 The proposed algorithm: IWODEMO

The IWODEMO algorithm proposes hybridization between the DE and IWO algorithms. A description of the proposed algorithm is outlined in Figure 1. The initial population POP_0 of size N will evolve over several iterations producing an exact number of N offspring. A solution

$\vec{X}_i = [x_{i,1}, \dots, x_{i,j}, \dots, x_{i,D}]$, often called vector, is represented by *D* decision variables. As in IWO, a solution produces a number of offspring in relation to its rank in the sorted population. Two factors are used for this purpose: a dominance factor (sorting by front as in NSGAII [1]) and an isolation factor (crowding distance). Once the population is sorted, the solution that occupies the first rank (the most isolated) produces the first offspring. Then, the following solutions produce descendants until *N* offspring are produced. At each iteration g, a solution $\vec{X}_i$ produces a number of offspring $ND_{i,g}$ proportional to its objective function value.

```
Create randomly an initial population $POP_0$ of size $N$ and evaluate each solution $\vec{X}_i$ on each objective
Sort $POP_0$ according to dominance and isolation factors
WHILE no stopping rule is invoked
    WHILE $NbOffspring < N$
        For each $ND_{i,g}$ offsprings of $\vec{X}_i$
            REPRODUCTION - Create a trial vector $\vec{U}_i$ with adaptive mutation DE/rand/1/bin
            SELECTION – 3 possible cases
                a) If $\vec{U}_i$ dominate $\vec{X}_i$, replace $\vec{X}_i$ by $\vec{U}_i$
                b) If $\vec{U}_i$ is dominated by $\vec{X}_i$, delete $\vec{U}_i$ and do WEED COLONIZATION
                c) If no dominance relationship, add $\vec{U}_i$ to $POP_g$
            WEED COLONIZATION – (Only if $\vec{U}_i$ is dominated)
                Plant a seed $\vec{W}_i$ according to IWO weed colonization: $\vec{w}_{i,j} = \vec{x}_{i,j} + N(0, \sigma_g^2)\ for\ j = 1, \dots D$     (1)
                Do SELECTION with $\vec{W}_i$ (instead of $\vec{U}_i$)
        Sort $POP_g$ according to dominance and isolation factors
        Delete identical solutions in $POP_g$ and progressive REDUCTION of $POP_g$ to $N$ solutions
Return non-dominated solutions
```

Figure 1. Outline of IWODEMO algorithm

Two types of offspring can be produced from a solution $\vec{X}_i$: the trial vectors $\vec{U}_i$ according to the DE algorithm and the seeds $\vec{W}_i$ according to the IWO algorithm. In the first case, a target vector $\vec{X}_i$ produces a trial vector $\vec{U}_i$ according to the adaptive mutation DE/rand/1/bin borrowed from the DE algorithm. It should be noted that the *CR* parameter is modified over the iterations allowing exploration at the beginning of the resolution and then to make more exploitation. The selection is then carried out with the two considered vectors $\vec{X}_i$ et $\vec{W}_i$. This consists of comparing the trial vector to the target vector. If he dominates the target vector, he replaces it. If no dominance is identified between the two solutions, the trial vector is added to the population as in DEMO algorithm [4]. If the trial vector is dominated, it is deleted. In the second case, IWODEMO produces a seed $\vec{W}_i$ from the plant $\vec{X}_i$ according to Eq. (1) in Figure 1 where *j* corresponds to the decision variables and $N(0, \sigma_g^2)$ corresponds to a random value derived from a normal distribution of mean zero and of variance equal to the standard deviation squared. This offspring creation corresponds to the principle of plant proliferation borrowed from the IWO algorithm except that a mutation probability is added to the equation: the disturbance is done on decision variables under mutation criterion. The two considered vectors, $\vec{X}_i$ et $\vec{W}_i$, then move to the selection. However if the seed is dominated by the target vector, it is removed and the algorithm continues without weed colonization.

The solution $\vec{X}_i$ continues the creation of its $ND_{i,g}$ offspring. However, it is possible that the solution is replaced by a descendant and in this case, the offspring also produces other descendant. In total, a maximum of *N* offspring are produced during an iteration, including trial vectors and seeds. We can remark that IWODEMO allows the immediate replacement of a solution which encourages a faster convergence. Following the creation of the offspring, identical solutions are deleted in POP_g. The population is sorted again and a progressive reduction ensures the maintenance of the *N* best solutions.

3 Numerical experiments and results

The performance of IWODEMO is evaluated using a comparative study with the main hybrid algorithms involving the DE algorithm for solving MO problems from the literature. More specifically, these algorithms are NSDE [11], DEMO [4] and ADE-MOIA [5]. To carry out numerical experimentations, the well-known ZDT test instances introduced by Zitzler, Deb and Thiele [12] are used. This benchmark is probably one of the most widely used references in the continuous MO optimization literature. ZDT instances contain two objective functions. The feasible solutions are in real encoding and have 30 (ZDT1, ZDT2, ZDT3) or 10 (ZDT4, ZDT6) decision variables. Pareto-optimal fronts contain an infinite bounded set of solutions. These instances also offer diversity in the solution space. Indeed, the Pareto-optimal fronts present particular forms thus leading to the complexity in solving of these instances.

In order to measure both the quality of the solutions as well as their diversity, the performance evaluation of the algorithms is realized using two metrics: the convergence metric γ and the diversity metric Δ. The Pareto-optimal front of each instance must be known for the computation of these metrics. Pareto-optimal fronts composed of 500 uniformly spaced solutions made available online by S. Huband are used (http://www.scis.ecu.edu.au/research/wfg/datafiles.html). The performance comparison with these different algorithms is not easy to achieve. Indeed, the original papers presenting these algorithms generally use a diversified implementation and a performance evaluation that can hardly be compared. In order to propose a reliable and fair comparative study, the algorithms NSDE, DEMO and ADE-MOIA from the literature were recoded and tested under the same experimental conditions.

The extended version of this paper will detail the experimental conditions as well as the obtained results. According to these results, it will be demonstrated that IWODEMO effectively combines the forces of ED and IWO that have been hybridized. Indeed, IWODEMO generally obtains better results than the other algorithms from the literature used in comparison in terms of convergence, dispersion of solutions on the Pareto-optimal front and computation time.

4 Conclusion

IWODEMO is an effective alternative for solving MO problems with continuous variables. In particular, these results highlight the importance of diversification and intensification mechanisms on the overall performance of an MO algorithm. The weed colonization phase borrowed from IWO algorithm and the adaptive parameters of IWODEMO seem to be a hybridization scheme complementary to the DE which is recognized to have a global search capability. It is also important to note that IWODEMO is a generic Pareto algorithm, so it can be extended and adapted to solve other MO problems. In the next steps of the research, it is therefore planned to test IWODEMO by increasing the number of objective functions. In addition, it would be interesting to test the effectiveness of the algorithm on problems from the industry.

References

[1] K. Deb, *et al.* (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Transactions on Evolutionary Computation, 6, p. 182-197.

[2] J. Holland (1975). *Adaptation in Natural and Artificial Systems*, Ann Harbor, University of Michigan Press.

[3] R. Storn, K. Price (1997). Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces, Journal of Global Optimization, 11, p. 341-359.

[4] T. Robic, B. Filipic (2005). DEMO: Differential Evolution for Multiobjective Optimization, in: A.H.A. Carlos A. Coello Coello, and Eckart Zitzler (Ed.) EMO, Springer.Lecture Notes in Computer Science Guanajuato, Mexico.

[5] Q. Lin, *et al.* (2015). A novel hybrid multi-objective immune algorithm with adaptive differential evolution, Computers & Operations Research, 62, p. 95-111.

[6] A.R. Mehrabian, C. Lucas (2006). A novel numerical optimization algorithm inspired from weed colonization, Ecological Informatics, 1, p. 355-366.

[7] I. Boussaïd, *et al.* (2013). A survey on optimization metaheuristics, Information Sciences, 237, p. 82-117.

[8] J.I. Fister, *et al.* (2013). A Brief Review of Nature-Inspired Algorithms for Optimization, arXiv preprint arXiv:1307.4186.

[9] D. Kundu, *et al.* (2011). Multi-objective optimization with artificial weed colonies, Information Sciences, 181, p. 2441-2454.

[10] A.H. Nikoofard, *et al.* (2012). Multiobjective invasive weed optimization: Application to analysis of Pareto improvement models in electricity markets, Applied Soft Computing, 12, 100-122.

[11] A. Iorio, X. Li (2005). Solving Rotated Multi-objective Optimization Problems Using Differential Evolution, in: S.B.H. G. Webb and X. Yu (Ed.) AI 2004: Advances in Artificial Intelligence, pp. p. 861-872.

[12] E. Zitzler, *et al.* (2000). Comparison of multiobjective evolutionary algorithms: Empirical results, Evolutionary Computation, 8, p. 173-195.

On the Sensitivity of Grid-Based Parameter Adaptation Method

Vasileios A. Tatsis and Konstantinos E. Parsopoulos

Department of Computer Science & Engineering,
University of Ioannina, GR-45110 Ioannina, Greece,
{vtatsis,kostasp}@cse.uoi.gr

Abstract. The rising popularity of metaheuristics has placed their parameter tuning in the center of research in the past decades. It is an important issue with significant implications on their overall performance, especially in demanding problems of high complexity. On the one hand, inappropriate parameters may render the algorithm incapable of detecting good quality solutions. On the other hand, parameter tuning through trial-and-error procedures expands the necessary experimentation time and consumes valuable computational resources. Recently, a general-purpose parameter adaptation method based on grid search in the parameter domain was proposed. The method was successfully demonstrated on two metaheuristics, namely Differential Evolution and Particle Swarm Optimization, attaining competitive performance against other methods. Similarly to other methods, the grid-based search has also a few user-defined parameters. The present work offers a first study of its sensitivity on these parameters. For this purpose, Differential Evolution is the tuned algorithm and the analysis is conducted on the established CEC 2013 test suite. The results verify previous evidence of the method's tolerance on its parameters.

1 Introduction

Metaheuristics have served as efficient solvers for many decades [1]. Among them, evolutionary algorithms have been distinguished for their effectiveness in a plethora of contemporary scientific applications [2, 3]. However, proper parametrization is usually the string attached to their promised success [4]. This deficiency has been tackled through parameter tuning methods, which are distinguished in offline and online methods.

Offline tuning requires a preprocessing phase where the employed metaheuristic is applied on a prescribed set of test problems. Appropriate parameter values are detected through a trial-and-error procedure where a number of different choices are considered and the algorithm's performance is assessed for each one on the test problems. Stochastic search algorithms require the iterative application of this procedure in order to extract statistically sound conclusions. Despite the obvious drawback of the high computational needs, offline approaches offer parameter values that can be reusable in similar problems. On the other hand, failing to select diverse test problems may results in over-specialization of the algorithm. Design of Experiments [5], F-Race [6], and ParamILS [7] are among the state-of-the-art in offline tuning methods.

Contrary to offline tuning, online approaches adapt the parameter setting of the algorithm during its execution, based on its performance during the specific experiment. Although they alleviate the over-specialization problem, they do not offer a single parameter set that can be used in different problem instances or algorithms. On the other hand, their clear advantage is less user intervention in the whole procedure. Concise reviews of online adaptation approaches can be found in [7, 8].

Recently, a general-purpose online parameter adaptation method was proposed in [9]. This method conducts grid search in the parameters' domain during the execution of the algorithm. The search is driven by estimations of the algorithm's neighboring parameter vectors and can be used both for categorical, integer, or real-valued parameters. The method was validated on the Differential Evolution algorithm, which is widely known for its sensitivity on its parameters, as well as on Particle Swarm Optimization [9–11]. The large-scale test problems of the test suite in [12] served as the corresponding testbed. Similarly to other approaches, the method involves a small number of parameters that offer tunability.

The present work constitutes a first study on the sensitivity of the grid-based parameter tuning method on its user-defined parameters. For this purpose, the Differential Evolution algorithm that offered interesting conclusions in previous works is adopted in the present study. Moreover, the established CEC 2013 test suite is considered as the corresponding testbed. Several levels of the basic parameters are considered and their influence on the algorithm's performance is statistically analyzed, offering interesting conclusions.

The rest of the paper is organized as follows: Section 2 briefly presents Differential Evolution, and the grid-based parameter adaptation method is described in Section 3. The experimental configuration for the sensitivity analysis along with the analysis of the results is offered in Section 4. Finally, the paper concludes in Section 5.

2 Differential Evolution

Differential Evolution (DE) [13] is a population-based metaheuristic. After two decades of ongoing development, it is currently considered among the state-of-the-art in evolutionary computation [14]. Given the unconstrained n-dimensional optimization problem

$$\min_{x \in X \subset \mathbb{R}^n} f(x),$$

DE utilizes a population of N search points,

$$P = \{x_1, x_2, \ldots, x_N\}.$$

Each population member is a candidate solution vector

$$x_i = (x_{i1}, x_{i2}, \ldots, x_{in})^\top \in X, \quad i = 1, 2, \ldots, N,$$

and it is randomly and uniformly initialized in the search space X.

The population is evolved through the iterative application of mutation, crossover, and selection procedures. Mutation consists of the generation of a new vector u_i for each member x_i of the population. The new vector is generated by adding a weighted difference of randomly selected members of the population on a base vector that varies from one mutation operator to another. The baseline mutation operators for x_i are as follows:

$$\text{DE/best/1:} \quad u_i = x_g + F\left(x_{r_1} - x_{r_2}\right), \tag{1}$$

$$\text{DE/rand/1:} \quad u_i = x_{r_1} + F\left(x_{r_2} - x_{r_3}\right), \tag{2}$$

$$\text{DE/current-to-best:} \quad u_i = x_i + F\left(x_g - x_i + x_{r_1} - x_{r_2}\right), \tag{3}$$

$$\text{DE/best/2:} \quad u_i = x_g + F\left(x_{r_1} - x_{r_2} + x_{r_3} - x_{r_4}\right), \tag{4}$$

$$\text{DE/rand/2:} \quad u_i = x_{r_1} + F\left(x_{r_2} - x_{r_3} + x_{r_4} - x_{r_5}\right), \tag{5}$$

where the index g denotes the best member of the population P in terms of function value, i.e.,

$$g = \arg \min_{i=1,\ldots,N} \{f(x_i)\},$$

and $r_1, r_2, \ldots, r_5$, are mutually different, randomly selected numbers between 1 and N that differ also from i. The scale factor F is a user-defined parameter that is crucial for the algorithm since it controls the magnitude of the mutations.

Mutation is followed by crossover, where a trial vector v_i is produced for each x_i by randomly selecting components from the original and the mutated vector. Binomial crossover is defined as follows:

$$v_{ij} = \begin{cases} u_{ij}, & \text{if } \texttt{rand}() \leqslant CR \text{ or } j = R_i, \\ x_{ij}, & \text{otherwise,} \end{cases} \tag{6}$$

where $j \in \{1,2,\ldots,n\}$; `rand()` is a uniform random number generator in the range $[0,1]$; $CR \in (0,1]$ is another user-defined parameter of the algorithm called the crossover rate; and $R_i \in \{1,2,\ldots,n\}$ is a randomly selected index (different for each x_i at each iteration). The crossover procedure is responsible for the amount of information inherited from the original and the mutated vector, while it ensures that at least one component of v_i comes from the mutated vector. An alternative is the exponential crossover, where x_i is initially copied into v_i. Then, a random component of v_i is selected and all subsequent components are replaced by those of the mutated vector u_i until a stochastic condition is satisfied.

The iteration of the algorithm is completed by selection, where each x_i is replaced by the produced trial vector v_i if it achieves better objective function value.The population iteratively evolves until a termination condition is satisfied and the best detected solution x_g is reported.

3 Grid-Based Parameter Adaptation Method

The grid-based parameter adaptation method was initially proposed in [9] and used for the online control of the scalar parameters F and CR of the Differential Evolution algorithm. The method was expanded in [10] for the online adaptation of the mutation operator. The method was further demonstrated for the Particle Swarm Optimization algorithm in [11]. The central idea is the discretization of the parameters' domain and the adaptation of the algorithm's parameter values to neighboring values in the corresponding grid, based on short-run estimations of its performance.

Let us make our description more concrete by considering the Differential Evolution algorithm and its two scalar parameters F and CR, along with two discretization steps λ_{CR} and λ_F for their domains, respectively. Also, let S_{CR}, S_F, be the corresponding discretized sets. Then, the grid is formed as follows:

$$\mathscr{G} = \{(CR,F);\, CR \in S_{CR}, F \in S_F\}\,.$$

The algorithm starts from a parameter vector in $\mathscr{G}$ (the central point is a reasonable choice), and the population is randomly initialized in the search space of the problem at hand. This population is also called the primary population and it is denoted as P_p. Similarly, its parameter pair is called the primary parameter pair and denoted as (CR_p, F_p). According to the suggestion in [9], P_p is evolved for a number of iterations,

$$t_p = \alpha \times n,$$

where $\alpha > 1$ is an integer and n stands for the problem dimension. In the original work [9], both parameters were considered in the domain $[0,1]$ with $\lambda_{CR} = \lambda_F = 0.1$, and $\alpha = 10$. After the t_p iterations, the primary population stops and the following three phases take place.

Phase I: Cloning

The primary parameter pair (CR_p, F_p) has eight neighboring parameter pairs in $\mathscr{G}$ that are defined as follows [9]:

$$CR' = CR_p + i\,\lambda_{CR}, \quad F' = F_p + j\,\lambda_F, \quad i,j \in \{-1,0,1\}, \tag{7}$$

where the case $i = j = 0$ corresponds to the primary parameter pair itself. For each one of these parameter vectors, a secondary population is defined by cloning the primary population. In [10], the method included also the adaptation of the mutation operator. In this case, four additional secondary populations, also called bridging populations, where defined by cloning the primary population with the primary parameter pair but different mutation operator from the ones defined in Eqs. (1)-(5).

Thus, after the cloning phase we obtain 13 secondary populations denoted as P_{s_j}, $j = 1,2,\ldots,13$, which are identical with the primary one, but 9 of them have the parameter pairs defined in Eq. (7) and same mutation operator with the primary population, while the remaining 4 populations have the primary parameter pair but a different mutation operator each.

Phase II: Performance Estimation

Each one of the 13 secondary populations is individually evolved for t_s iterations in order to reveal its dynamic with the new parameters. Typically, t_s shall be significantly smaller than t_p to spare computational resources (function evaluations). The performance of the secondary populations can be measured using various performance measures. In [9] the average objective value (AOV) of the population was used, which is defined as follows:

$$\mathrm{AOV}_j = \frac{1}{N}\sum_{i=1}^{N} f(x_i), \quad x_i \in P_{s_j}, \quad i = 1,2,\ldots,N, \quad j = 1,2,\ldots,13. \tag{8}$$

This measure reveals the average improvement of the corresponding secondary population with its new parameters. In addition, the objective value standard deviation (OVSD) was also considered in [10], which is defined as follows:

$$\mathrm{OVSD}_j = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(f(x_i) - \mathrm{AOV}_j\right)^2}, \quad x_i \in P_{s_j}, \quad i = 1,2,\ldots,N, \quad j = 1,2,\ldots,13. \tag{9}$$

This performance measure determines the diversity of the population's values, which is desirable to alleviate rapid convergence in local minima.

The secondary populations compete using either AOV or both AOV and OVSD (in terms of Pareto dominance when more than one measure is used) and the best one is distinguished. If there are more than one dominant secondary populations, one is selected at random among them.

Phase III: Dynamic's Deployment

The selected secondary population is evolved for t_p iterations to fully reveal its dynamic with the selected parameters. Then, if its AOV improves the AOV of the primary population for at least $\varepsilon \geqslant 0$, the evolved population along with its parameters replaces the primary population. Otherwise, the primary population remains unaltered.

This step completes a full cycle of the method, and the whole procedure is repeated anew from the cloning phase. The maximum number of cycles can be predefined according to the available computational budget [9]. Pseudocode of the method, which is called Differential Evolution with Grid-based Parameter and Operator Adaptation (DEGPOA) is provided in Algorithm 1.

4 Performance Analysis

The DEGPOA algorithm was validated on high- and low-dimensional test suites in [9, 10] with promising results. Without any parameter tuning, the algorithm was capable to compete against other algorithm, controlling its parameters throughout the search procedure. Regarding the grid search parameters, the following values were proposed as default choices [9]:

$$t_s = 5, \quad t_p = 10 \times n, \quad \lambda = \lambda_F = \lambda_{CR} = 0.1, \tag{10}$$

while the initial parameter vector was placed at the center of the grid, i.e., $(CR_p, F_p) = (0.5, 0.5)$, and the initial primary operator was randomly selected from the ones in Eqs. (1)-(5). Thus, the main effect of DEGPOA's parameters remains to be studied.

In the present work we considered three sets of parameter values for t_s, t_p, and λ:

$$S_{t_s} = \{5, 10, 15, 20\}, \quad S_{t_p} = \{5 \times n, 10 \times n, 15 \times n, 20 \times n\}, \quad S_\lambda = \{0.01, 0.05, 0.1, 0.2\}.$$

The previously used version with the parameters of Eq. (10) was considered as the baseline for assessing the new grid search settings. DEGPOA was validated by changing one of its parameters to a different level from the sets above, while keeping the rest of the parameters fixed to the baseline values. This results in 12 new DEGPOA instances.

All experiments were conducted on the established CEC 2013 test suite [15]. This suite consists of 28 unimodal, multimodal, and composite functions, henceforth denoted as f_1-f_{28}. The search space for all test problems is $[-100, 100]^n$, where n stands for the dimension. We considered the

Algorithm 1 DEGPOA: Differential Evolution with Grid-based Parameter and Operator Adaptation

```
INITIALIZE($P, F_p, CR_p, O_p$)
EVOLVE($P, F_p, CR_p, O_p, t_p$)
$m \leftarrow 13$
while (NOT TERMINATION) do
    /* Phase I: Cloning */
    for ($i = 1 : m$) do
        if ($i \leqslant 9$) then
            /* Secondary population: same operator, different parameters (use Eq. (7)) */
            $(P_{s_i}, F_i, CR_i, O_i) \leftarrow (P, F', CR', O_p)$
        else
            /* Bridging secondary population: different operator, same parameters */
            $(P_{s_i}, F_i, CR_i, O_i) \leftarrow (P, F, CR, O'_i)$
        end if
    end for
    /* Phase II: Performance Estimation */
    for ($i = 1 : m$) do
        EVOLVE($P_{s_i}, F_i, CR_i, O_i, t_s$)
    end for
    /* Phase III: Dynamic's Deployment */
    $(P_{\text{best}}, F_{\text{best}}, CR_{\text{best}}, O_{\text{best}}) \leftarrow$ SELECT_BEST($P_{s_i}, F_i, CR_i, O_i, \text{AOV}, \text{OVSD}$)
    EVOLVE($P_{\text{best}}, F_{\text{best}}, CR_{\text{best}}, O_{\text{best}}, t_p$)
    /* Update primary population */
    if $\left(\text{AOV}_P - \text{AOV}_{P_{\text{best}}} \geqslant \varepsilon\right)$ then
        $(P, F_p, CR_p, O_p) \leftarrow (P_{\text{best}}, F_{\text{best}}, CR_{\text{best}}, O_{\text{best}})$
    end if
end while
```

most common cases $n = 10$ and $n = 30$ in our study. Also, the guidelines of the test suite dictate that the maximum computational budget is

$$T_{\max} = 10^4 \times n,$$

while the performance is measured by using the error gap between the known optimal solution of the problem, x_{opt}, and the solution x^* achieved by the algorithm,

$$\varepsilon^* = f\left(x^*\right) - f\left(x_{\text{opt}}\right).$$

In order to avoid any bias imposed by the initial parameter set, the central parameter $(CR_p, F_p) = (0.5, 0.5)$ was used in all cases. Note that according to the CEC 2013 requirements, a fixed population size $N = 60$ was used and 51 independent experiments were conducted per problem.

Henceforth, we denote as DEGPOA$_{\text{base}}$ the baseline version of the algorithm, and the rest are denoted with corresponding subscripts. For example, the instance with $t_s = 5$, $t_p = 5 \times n$, and $\lambda = 0.01$, is denoted as DEGPOA$_{5s\_5p\_0.01\lambda}$.

Table 1. Comparisons of new DEGPOA instances with DEGPOA$_{\text{base}}$.

	n	W	L	D	W-L	I	NI
t_s modified							
DEGPOA$_{5s\_10p\_0.1\lambda}$	10	0	0	28	0	28	1.00
DEGPOA$_{5s\_10p\_0.1\lambda}$	30	0	0	28	0	28	1.00
DEGPOA$_{10s\_10p\_0.1\lambda}$	10	1	5	22	-4	24	0.86
DEGPOA$_{10s\_10p\_0.1\lambda}$	30	1	2	25	-1	27	0.96
DEGPOA$_{15s\_10p\_0.1\lambda}$	10	0	11	17	-11	17	0.61
DEGPOA$_{15s\_10p\_0.1\lambda}$	30	2	3	23	-1	27	0.96
DEGPOA$_{20s\_10p\_0.1\lambda}$	10	1	12	15	-11	17	0.61
DEGPOA$_{20s\_10p\_0.1\lambda}$	30	4	7	17	-3	25	0.89
				absolute sum:	31		
t_p modified							
DEGPOA$_{5s\_5p\_0.1\lambda}$	10	0	4	24	-4	24	0.86
DEGPOA$_{5s\_5p\_0.1\lambda}$	30	0	2	26	-2	26	0.93
DEGPOA$_{5s\_10p\_0.1\lambda}$	10	0	0	28	0	28	1.00
DEGPOA$_{5s\_10p\_0.1\lambda}$	30	0	0	28	0	28	1.00
DEGPOA$_{5s\_15p\_0.1\lambda}$	10	1	1	26	0	28	1.00
DEGPOA$_{5s\_15p\_0.1\lambda}$	30	1	0	27	1	29	1.04
DEGPOA$_{5s\_20p\_0.1\lambda}$	10	2	1	25	1	29	1.04
DEGPOA$_{5s\_20p\_0.1\lambda}$	30	3	1	24	2	30	1.07
				absolute sum:	10		
λ modified							
DEGPOA$_{5s\_10p\_0.05\lambda}$	10	1	3	24	-2	26	0.93
DEGPOA$_{5s\_10p\_0.05\lambda}$	30	1	2	25	-1	27	0.96
DEGPOA$_{5s\_10p\_0.1\lambda}$	10	0	0	28	0	28	1.00
DEGPOA$_{5s\_10p\_0.1\lambda}$	30	0	0	28	0	28	1.00
DEGPOA$_{5s\_10p\_0.15\lambda}$	10	0	2	26	-2	26	0.93
DEGPOA$_{5s\_10p\_0.15\lambda}$	30	2	0	26	2	30	1.07
DEGPOA$_{5s\_10p\_0.2\lambda}$	10	0	6	22	-6	22	0.79
DEGPOA$_{5s\_10p\_0.2\lambda}$	30	4	7	17	-3	25	0.89
				absolute sum:	16		

The twelve new DEGPOA instances were tested on the CEC 2013 test suite according to the settings above, and their results were recorded and statistically analyzed in order to facilitate comparisons with DEGPOA$_{\text{base}}$. For this purpose, Wilcoxon significance tests at confidence level 95% were used to compare the achieved solution errors. For each comparison of a new instance with the baseline variant, a win was counted if it achieved statistically superior performance than the baseline approach. In the opposite case, a loss was counted, while statistically insignificant differences between algorithms were considered as ties.

Table 1 report the number of wins (denoted as "+"), losses (denoted as "−"), and ties (denoted as "=") of the new DEGPOA instances against DEGPOA$_{\text{base}}$. The fourth column denoted as "W-L" stands for the difference between the number of wins and loses, which provides the general performance trend of the corresponding new instance against the baseline. High positive values correspond to an instance that has far better performance than the baseline, while negative values imply inferior performance of the new instance. The next column denoted as I reports the index value

$$I = 28 + (W - L),$$

which characterizes the relevant performance of the corresponding DEGPOA instance against the baseline over all the 28 test problems. The last column of the table denoted as NI is the normalized index,

$$NI = \frac{\text{Index}}{28},$$

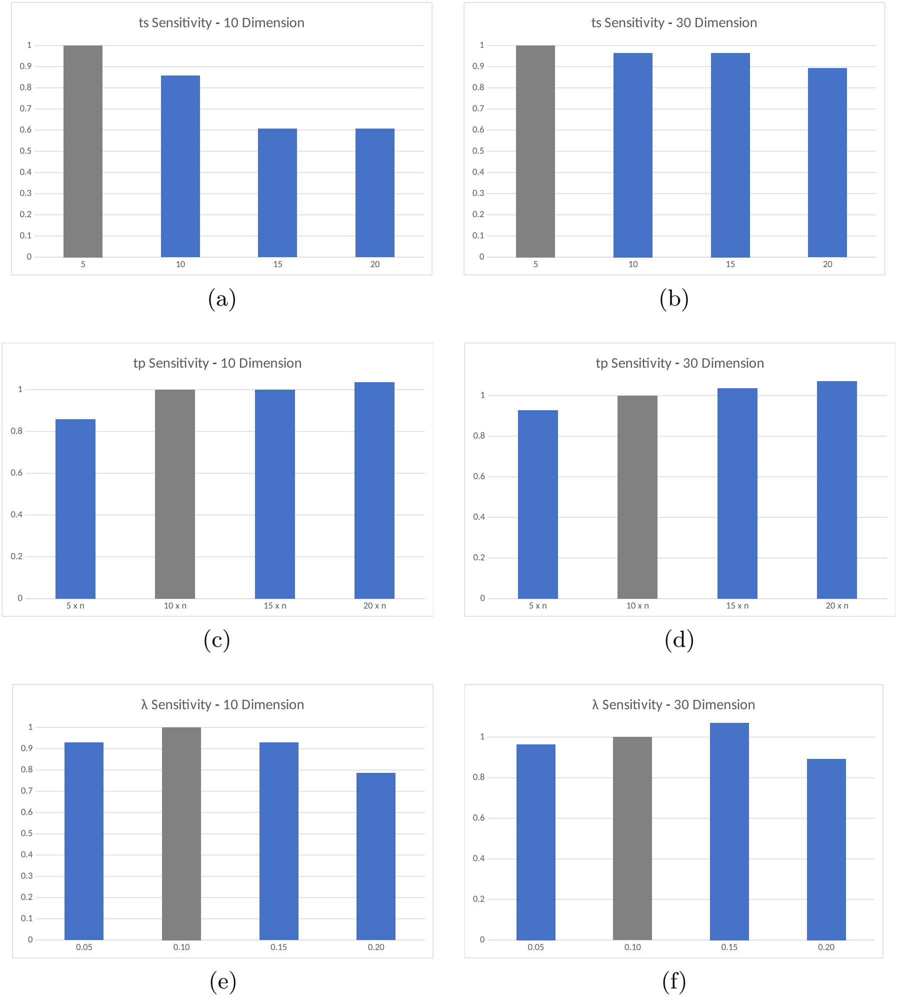


Fig. 1. Values of the normalized index NI per dimension for the parameters t_s (cases (a) and (b)), t_p (cases (c) and (d)), and λ (cases (e) and (f)).

which offers a straightforward comparison measure between the competing algorithms (rounded to 2 decimal digits). Obviously, $NI = 1.00$ when the two compared algorithms have statistically equivalent performance (only ties in statistical tests), while it becomes $NI > 1.00$ whenever the new DEGPOA instance is superior than the baseline, and $NI < 1.00$ when it is inferior. Since $0 \leqslant I \leqslant 56$, the normalized index is bounded in $0 \leqslant NI \leqslant 2$.

In order to facilitate comparisons, Fig. 1 illustrates NI for the different parameter level and dimension. The gray bar stands for the performance of DEGPOA$_{\text{base}}$ while the blue bars refer to the corresponding new instances. The figures offer some interesting conclusions. Firstly, we can observe that t_s can have significant impact on the algorithm's performance in lower dimension ($n = 10$) as we can see in Fig. 1(a). Specifically, smaller values of t_s offer better overall performance, which implies that the estimations of the secondary populations are adequately accurate, sparing computational budget for the dynamic's deployment phase. On the other hand, in the higher-dimensional case ($n = 30$) depicted in Fig. 1(b) this effect becomes milder as a direct consequence of the increased complexity of the problems, which requires longer estimation runs. Nevertheless, the value $t_s = 5$ that was used in previous works [9] verifies its superiority for the specific dimensions.

Regarding the parameter t_p, we can observe in Figs 1(c) and 1(d) that values lower than $10 \times n$ produce inferior performance, implying that the number is inadequate to reveal the primary population and parameters' dynamic. Instead, higher values are beneficial especially for the high-dimensional case. However, the effect remains bounded within 10% of the corresponding baseline value even after doubling the value of t_p. This indicates that the effect of t_p is not highly for the

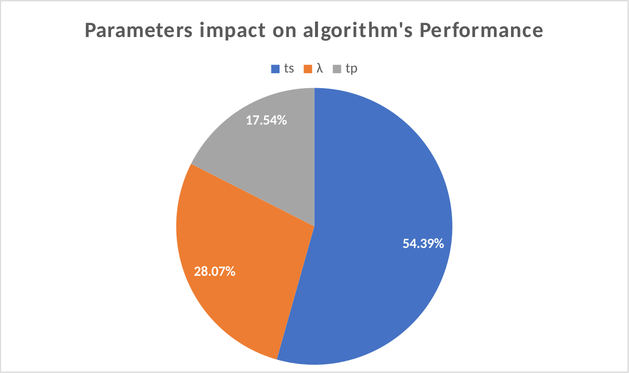


Fig. 2. Overall Parameters impact on algorithm's performance

algorithm's performance if the estimation evaluations t_s retain a proper value. Recall that in all experiments for different t_p values, the default (sub-optimal) $t_s = 5$ value was used.

For both t_s and t_p, the main performance pattern (improving or worsening) was observed for both dimensions. However, this is not the case for the third parameter λ. Changing the discretization step from 0.1 to either lower or higher values produces inferior performance in the 10-dimensional case as illustrated in Fig. 1(e). This motif changes in the higher-dimensional case as illustrated in Fig. 1(f), where slightly increasing λ to 0.15 improves performance for 7%, while different values produce inferior performance of comparable magnitude. Notice that λ determines the search accuracy in the parameter space and has actual dependence both on the algorithm as well as the problem at hand. Thus, there is no clear explanation for this behavior, which is probably the outcome of the interplay between the algorithm's dynamic with the specific parameters and the complexity of the problem itself.

The results show that DEGPOA can achieve stable performance under mild perturbations of the proposed default parameters. In order to identify the overall most influential parameter for all DEGPOA instances, we considered the sum of the absolute differences $W - L$ for each parameter as they are reported in Table 1. Then, we normalized these three values by dividing with their sum, and we received the percentages that are graphically represented in Fig 2. Each normalized value shows the participation of the corresponding parameter in the observed differences. The blue color refers to the t_s parameter, which proves to be the most influential one, followed by λ and t_p.

5 Conclusions

The rising popularity of metaheuristics has placed their parameter tuning in the center of research in the past decades. It is an important issue with significant implications on their overall performance, especially in demanding problems of high complexity. On the one hand, inappropriate parameters may render the algorithm incapable of detecting good quality solutions. On the other hand, parameter tuning through trial-and-error procedures expands the necessary experimentation time and consumes valuable computational resources. Recently, a general-purpose parameter adaptation method based on grid search in the parameter domain was proposed. The method was successfully demonstrated on two metaheuristics, namely Differential Evolution and Particle Swarm Optimization, attaining competitive performance against other methods. Similarly to other methods, the grid-based search has also a few user-defined parameters. The present work offers a first study of its sensitivity on these parameters. For this purpose, Differential Evolution is the tuned algorithm and the analysis is conducted on the established CEC 2013 test suite. The results verify previous evidence of the method's tolerance on its parameters.

Metaheuristics are part of the state-of-the-art in optimization literature for solving demanding problems. However, their performance is strongly dependent on their parameters. This deficiency has resulted in a variety of parameter adaptation methods. Most of them constitute ad-hoc procedures designed for a specific algorithm.

The present work offered a first study of the sensitivity of the recently proposed grid-based parameter adaptation method on the mainstream CEC 2013 test suite. The method was previously validated on the Differential Evolution and Particle Swarm Optimization algorithm for the online control of their parameters. The analysis reveals that the performance estimation phase is the most sensitive one, while the rest of the parameters have only mild influence on the algorithm's dynamic. Also, it reveals that the previously proposed default parameters are very efficient.

Future work will expand the analysis in order to reveal possible interactions between the parameters by using methodologies such as the analysis of variance.

References

1. Gendreau, M., Potvin, J.: Handbook of Metaheuristics, 2nd edn. Springer New York Dordrecht, Heidelberg London (2010)
2. Gogna, A., Tayal, A.: Metaheuristics: review and application. Journal of Experimental & Theoretical Artificial Intelligence 25 (2013) 503–526
3. Torres-Jiménez, J., Pavón, J.: Applications of metaheuristics in real-life problems. Progress in Artificial Intelligence 2 (2014) 175–176
4. Eiben, A.E., Hinterding, R., Michalewicz, Z.: Parameter control in evolutionary algorithms. IEEE Transactions on Evolutionary Computation 3 (1999) 124–141
5. Bartz-Beielstein, T.: Experimental Research in Evolutionary Computation: The New Experimentalism. Springer (2006)
6. Birattari, M.: Tuning Metaheuristics: A Machine Learning Perspective. Springer (2009)
7. Hoos, H.H.: Automated algorithm configuration and parameter tuning. In Hamadi, Y., Monfroy, E., Saubion, F., eds.: Autonomous Search. Springer, Berlin Heidelberg (2011) 37–72
8. Eiben, A.E., Smit, S.K.: Evolutionary algorithm parameters and methods to tune them. In Hamadi, Y., Monfroy, E., Saubion, F., eds.: Autonomous Search. Springer, Berlin Heidelberg (2011) 15–36
9. Tatsis, V.A., Parsopoulos, K.E.: Differential evolution with grid-based parameter adaptation. Soft Computing 21 (2017) 2105–2127
10. Tatsis, V.A., Parsopoulos, K.E.: Grid search for operator and parameter control in differential evolution. In: Proceedings of the 9th Hellenic Conference on Artificial Intelligence. SETN '16, New York, NY, USA, ACM (2016) 7:1–7:9
11. Tatsis, V.A., Parsopoulos, K.E.: Grid-based parameter adaptation in particle swarm optimization. In: Proceedings of the 12th Metaheuristics International Conference (MIC 2017). (2017)
12. Lozano, M., Herrera, F., Molina, D.: Scalability of evolutionary algorithms and other metaheuristics for large-scale continuous optimization problems. Soft Computing 15 (2011) 2085–2087
13. Storn, R., Price, K.: Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. J. Global Optimization 11 (1997) 341–359
14. Das, S., Suganthan, P.N.: Differential evolution: A survey of the state-of-the-art. IEEE Transactions on Evolutionary Computation 15 (2011) 4–31
15. : Complementary material: Special session & competition on real-parameter single objective optimization at cec-2013. (`http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC2013/CEC2013.htm`)

Application of fuzzy smoothing filter in empirical copula function

A. Kresta[1]

1. Faculty of Economics, Technical University of Ostrava, Sokolská tř. 33, 702 00 Ostrava
ales.kresta@vsb.cz

Keywords : dependence modeling, fuzzy smoothing filter, empirical copula, smoothing.

1 Introduction

In portfolio optimization problem, when modelling the joint cumulative distribution function, it is important to choose the accurate approach for dependence modelling. The most flexible tool for dependence modelling are copula functions, which can be, with some simplifications, divided into the following groups: elliptical copula functions based on chosen elliptical probability distributions and Archimedean copula functions based on chosen generators. The choice of concrete Elliptical or Archimedean copulas requires the knowledge about the type of dependence structure. There exist also empirical copula function, introduced by Deheuvels (1981), and kernel-based approach applicable to a copula setup.

In our paper, we propose to apply smoothing filter based on discrete fuzzy transform in order to smooth the empirical copula function. The smoothing of empirical copula function consists of applying discrete direct fuzzy transform followed by inverse fuzzy transform.

2 Discrete fuzzy transforms

In this section will focus on the technique of fuzzy transform originally proposed by Perfilieva (2004, 2006) as a tool for image processing. Recently, in Holčapek and Tichý (2010, 2011) it was suggested to use this method for financial time series smoothing.

Following Perfilieva (2006), when we apply fuzzy transform approach, as we can guess from the term fuzzy, the independent variables are fuzzyfied according to the proximity to a given point, which can be compared to the concept of weights obtained via probability distribution function in kernel regression approach. Next, the observations of the dependent variable are averaged, which forms a functional relation for a given point. This step of fuzzy transform is called direct fuzzy transform. Obviously, the second step is inverse fuzzy transform, within which we return back to the original crisp space and obtain a smoothed function describing the relation of both variables.

Assuming that R is a real interval, g is a finite real function given at the nodes $x_1 < \cdot\cdot\cdot < x_n$ with Dom(g) ⊆ R and $A = \{A_i | i \in I\}$ is a fuzzy r-partition of R determined by (T, S) such that Dom(g) is sufficiently dense with respect to A, one can say that a collection of real numbers $\{F_i | i \in I\}$ is discrete (direct) fuzzy transform of g with respect to A, if

$$F_i = \frac{\sum_{j=1}^{n} g(x_j) A_i(x_j)}{\sum_{j=1}^{n} A_i(x_j) h}. \quad (1)$$

The numbers F_i are called components of the discrete F-transform. The F-transform of an original function serves as its discrete representation, which can be e.g. successfully used in numerical computations. To bring the F-transform back we use the inverse F-transform,

$$g(x_j) = \sum_{i=1}^{n} F_i A_i(x_j). \quad (2)$$

The above mentioned functions and the whole concept can be easily generalized to higher dimensions, see e.g. Martino et al. (2011). This can be helpful when smoothing the empirical cumulative density function (cdf) or empirical copula function, which represents only the dependence structure in cdf.

3 Copula functions

Copula function is in the fact a real function, which maps the dependency among particular distribution functions into [0,1],

$$C:[0,1]^n \rightarrow [0,1] \text{ on } R^n, \quad (3)$$

Every copula function has to satisfy these three conditions: (i) $C(u)=0$ whenever $u=[0,1]^n$ has at least one component equal 0, (ii) $C(u)=u_i$ whenever $u=[0,1]^n$ has all the components equal to 1 except the i-th, which is equal to u_i and finally (iii) $C(u)$ is n-increasing.

Actually, any copula function can be regarded as a multidimensional distribution function with marginals in the form of standardized uniform distribution. Assume potentially dependent random variables with marginal distribution functions F_1, …, F_n and joint distribution function $F_{1,\dots,n}$. Then following the Sklar's theorem (Sklar, 1959):

$$F_{1,\dots,n}(x_1, \dots, x_n)=C(F_1(x_1), \dots, F_n(x_n)). \quad (4)$$

If all distribution functions are continuous, a copula function C is unique. Sklar's theorem implies also an inverse relation,

$$C(u_1, \dots, u_n)= F_{1,\dots,n}(F_1^{-1}(u_1), \dots, F_n^{-1}(u_n)). \qquad (5)$$

When we have m empirical observations $\left(x_1^m,...,x_n^m\right)$, the empirical copula function can be defined as follows,

$$C\left(u_1,....,u_n\right)=\frac{1}{m}\sum_{i=1}^{m}1\Big(\tilde{\iota} \qquad \tilde{} \qquad , . \qquad (6)$$

However, the above function is not smooth and less data we observe (m) or higher is the dimension (n) of the problem the more choppy the function is. In this problem the fuzzy smoothing filter can be applied.

4 Real data application

In the real world application, we assume daily returns of two important American stock market indices, DJIA and S&P 500 over the last year (19.6.2017 – 18.6.2018), see Figure 1. In Figure 2 we show the contour plot of empirical copula function (left) and copula function obtained by applying fuzzy smoothing filter (right).

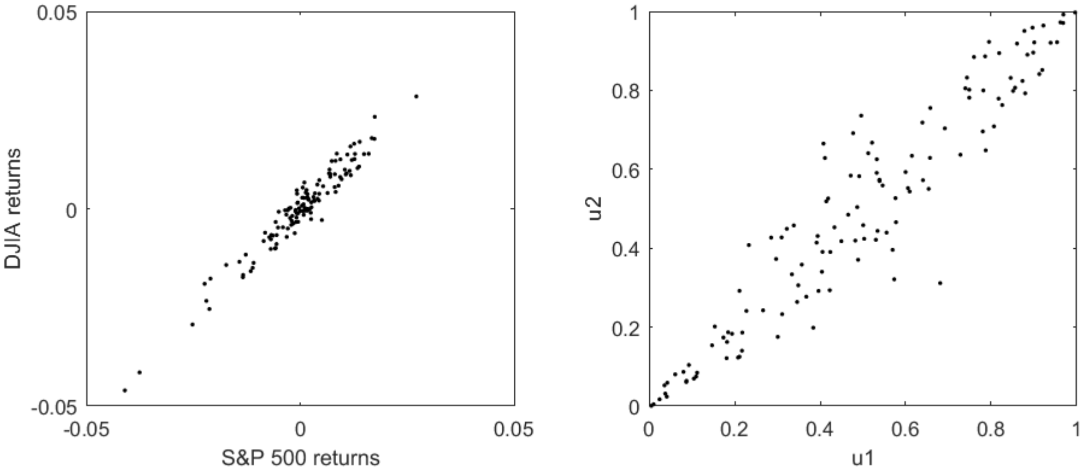


Figure 1: Daily returns (left) transformed to uniform distributions (right)

The advantage of the smoothed copula is also that it is specified by less parameters. In our application, we assumed 121 nodes (11 for each dimension). From these data (i.e. 121 values) we can obtain function value $C(u_1,u_2)$ for every $u \in [0,1]^n$. On the other hand, the empirical copula function is specified by 252 daily returns. As we can see, by smoothing the empirical copula function there is less parameters.

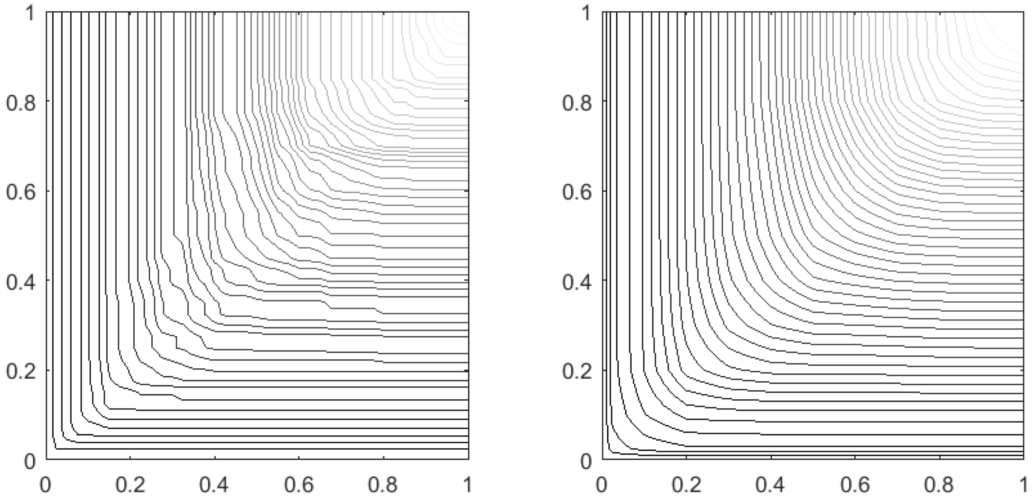


Figure 2: Contour plot of empirical copula function (left) smoothed by FS filter (right)

Moreover, we compare the fitness of smoothed copula function to the fitness of well-known parametrical copulas. In Table 1, we depict the mean absolute difference between empirical copula and smoothed copula, the benchmark is the mean absolute difference between empirical copula and estimated parametrical copulas. As it is apparent from the table, the smoothed copula has the lowest mean absolute error (approximately the half of the mean absolute error of the parametrical copulas). However, it must be said that the parametrical copulas have only one parameter (all copulas except Student copula) or two parameters (Student copula).

Table 1: Mean absolute error of smoothed empirical copula and parametrical copulas

Copula	FS filter	Gaussian	Student	Clayton	Frank	Gumbel
MAE	0,008	0,015	0,015	0,015	0,017	0,015

Acknowledgements

The research is supported by an SGS project of VSB-TU Ostrava under No. SP2018/34 and the Czech Science Foundation through project No. GA18-13951S. The support is greatly acknowledged.

References

[1] Deheuvels, P. (1981). *A Nonparametric Test for Independence*. Universit´e de Paris, Institut de Statistique.

[2] Holčapek, M., Tichý, T. (2010). A probability density function estimation using F-transform. *Kybernetika*, 46(3), 447– 458.

[3] Holčapek, M., Tichý, T. (2011). A smoothing filter based on fuzzy transform. *Fuzzy Sets and Systems*, 180(1), 69–97.

[4] Martinoa, F., Loiab, V., Sessaa, S. (2011). Fuzzy transforms method in prediction data analysis. *Fuzzy Sets and Systems*, 180(1), 146–163.

[5] Perfilieva, I. (2006). Fuzzy transforms: Theory and applications. *Fuzzy Sets and Systems*, 157(8), 993–1023.

[6] Perfilieva, I. (2004). Fuzzy transforms. Peters, James F. (ed.) et al. *Transactions on Rough Sets II. Rough sets and fuzzy sets*. Berlin: Springer.

Machine–learning algorithms for portfolio optimization problems

Noureddine Kouaissah[1] and Sergio Ortobelli[1]

[1] University of Bergamo, Department of MEQM, Via dei Caniana 2, 24127 Bergamo, Italy
nouredine.kouaissah@unibg.it

Abstract. In this paper, we investigate the use of machine–learning algorithms in portfolio selection problems using high–frequency data from S&P 500 index. In particular, we propose a heuristic method based on neural networks for optimal portfolio selection, which integrates different forecasts and trading strategies as well as investors preferences. We consider new performance measures for heavy–tailed distributions, and we discuss the dimensionality reduction problems. Ex-Ante and ex-post empirical analyses show that the proposed methods are fast and efficient to approximate the returns in large-scale portfolio problems. The proposed models are certainly expected to be promising approaches in the portfolio strategies where data is heavy-tailed and non–linear, whose patterns are difficult to be captured by traditional models.

1 Introduction

In recent decades, there has been increasing interest in the machine–learning algorithms for stock prices predictability (see, e.g., Sermpinis et al. (2013), Barak et al. (2017), and Krauss et al. (2017)). The results are encouraging and suggest that learning methods may be a useful device for both understanding and building profitable strategies. Therefore, we implement and analyze the usefulness of dynamic artificial neural network, hybrid neural networks, and several ensembles of these methods in the context of portfolio strategies; especially in a high–frequency environment. Accurate forecasting is the key element for better financial decision–making. In financial literature, many artificial neural network models are evaluated against statistical technique for forecasting the stock prices (see, e.g., Guresen et al. (2011)). However, there are very few studies that attempt to consider the following crucial aspects all together: portfolio selection problems, dimensionality reduction, investors preferences, and neural network algorithms using high–frequency databases.

Portfolio selection problems can be characterized and classified based on the motivations and intentions of investors (see Ortobelli et al. 2017). Thus, it is important to classify the optimal choices for any admissible ordering of preferences. In this regard, several classifications of the reward–risk probability function consistent with different orderings have been proposed in the financial literature (see, among others, Stoyanov et al. 2007; Ortobelli et al. 2017). We recall that an investor is risk-averse (risk-seeker) if he/she has concave (convex) utility function. In contrast to classical results on expected utility theory (von Neumann and Morgenstern (1944)), many empirical and theoretical studies support dominance rules of behavioral finance (see, among others, Levy and Levy (2002), Ortobelli et al. (2009) and the references therein) suggesting that investors prefer more than less and are neither risk averters nor risk seekers.

According to many researchers (e.g., Rachev et al. (2005)), the portfolio dimensional problem is strongly related to the estimation of the statistical input parameters, which describe the dependence structure of the returns. In this context, aiming to get a good approximation of the portfolio reward-risk measures, Papp et al. (2005) and Kondor et al. (2007), among others, have shown that the number of observations should increase with the number of assets. Therefore, in order to obtain a sound approximation of portfolio input measures, it is important to find the right trade-off between the number of historical data observations and the quality of statistical approximation of the historical observations that depend only on a few parameters. Notwithstanding, many studies illustrate that the problem of parameter uncertainty increases with the number of assets (see Kan and Zhou, 2007).

In practice, many methods have been proposed to predict stock trends. Initially, classical regression techniques were used to predict stock trends. Since stock data can be categorized as non–stationary time series, non-linear machine–learning techniques have also been suggested (see, Patel et al. (2015)). Artificial neural network and hybrid neural networks are machine learning algorithms which are widely used for predicting stock prices (see, e.g., Rather et al. (2015), Gocken et al. (2016)). These algorithms for pattern recognition emulate functioning of our brain to learn by creating a network of neurons.

In this paper, we integrate machine–learning algorithms with those of portfolio strategies to consider heavy tails of the return distributions, dimensionality reduction, and accurate prediction. An alternative would be to use an approach based on non–parametric techniques that have received significant interest from academics and the investment management community (see, e.g., Scott (2015), and Kouaissah et al. (2018)). The next section outlines the crucial aspects of the proposed methods.

2 Method description

As mentioned above, our aim is to build trading strategies that benefit from the efficient machine–learning algorithms, account for different investors preferences and use high–frequency data. Therefore, we define the following crucial aspects that are useful for the proposed methodologies:

- a description of different correlation measures (that represent the dependence structure between random variables) used to identify the main factors that consider all return variability (through PCA);
- a description of alternative methodologies to approximate the dependence between returns and factors;
- a heuristic method based on neural networks, and new performance measures, for optimal portfolio selection;
- an ex-post and an ex-ante analyses to discuss the impact of using different machine–learning algorithms, correlation measures and approximation methods.

Thus, we theoretically and empirically compare different methodologies based on approximating the dependence between returns and factors obtained from a PCA. The main contribution of this paper is the introduction of machine learning techniques to the portfolio theory. Focused improvements create sound trading strategies that are consistent with investors preferences, benefit from dimensionality reduction, and implement machine learning algorithms, involving the application of heuristic methods.

3 Conclusion

Portfolio selection problems often involve unknown parameters that have to be properly approximated from the data. Therefore, in this paper, we consider the implications of the machine–learning algorithms within the portfolio theory. We discuss and examine the impact of the correlation matrices, approximation methods, and investor's preferences in the portfolio theory; especially when we use high–frequency data. We use new performance measures that account for the heavy-tailed distribution of the returns and their joint risk. Finally, the proposed empirical analysis support the significance of the machine–learning algorithms within the portfolio theory.

Acknowledgements

This paper was supported by the Italian funds MURST 2016/2017 and by STARS Supporting Talented Research - Action 1 - 2017. The research was also supported by the Czech Science Foundation (GACR) under project 17-19981S, and by VSB-TU Ostrava under the SGS project SP2017/32.

References

1. Barak, S., Arjmand, A., Ortobelli, S. (2017). Fusion of multiple diverse predictors in stock market, *Information Fusion* **36**: 90–102.
2. Gocken, M., Ozcalici, M., Boru, A., Dosdogru, A.T. (2016). Integrating metaheuristics and Artificial Neural Networks for improved stock price prediction. *Expert SystemsWithApplications*, **44**, 320–331.

3. Guresen, K., Kayakutlu, G., Daim, T.U. (2011). Using artificial neural network models in stock market index prediction. *Expert Systems with Applications*, **38**, 10389–10397.
4. Kan, R., Zhou, G. (2007). Optimal portfolio choice with parameter uncertainty. *Journal of Financial and Quantitative Analysis*, **42**(3), 621–656.
5. Kondor, I., Pafka, S., Nagy, G. (2007). Noise sensitivity of portfolio selection under various risk measures. *Journal of Banking and Finance*, **31**, 1545–1573.
6. Kouaissah, N., Ortobelli, S., Tichy, T. (2018). Portfolio Theory and Conditional Expectations: Selected models and applications. *Ostrava: SAEL, Vol. 33, VSB-TU Ostrava*, Ostrava.
7. Krauss, C., Do, X.A., Huck, N. (2017). Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500. *European Journal of Operational Research* **259**: 689–702.
8. Levy, M., Levy, H. (2002). Prospect theory: Much ado about nothing? *Management Science* **48**(10): 1334–1349.
9. Ortobelli, S., Kouaissah, N., Tichy, T. (2017). On the impact of the conditional expectation in the portfolio theory. *Computational Management Science* **14**(4): 535–557.
10. Ortobelli, S. Rachev S., Shalit H., Fabozzi F. (2009) Orderings and probability functionals consistent with preferences. *Applied Mathematical Finance* **16**(1): 81–102.
11. Papp, G., Pafka, S., Nowak, M.A., Kondor, I. (2005). Random matrix filtering in portfolio optimization. *ACTA Physica Polonica B*, **36**, 2757–2765.
12. Patel, J., Shah, S., Thakkar, P., Kotecha, K. (2015). Predicting stock and stock price index movement using Trend Deterministic Data Preparation and machine learning techniques. *Expert Systems with Applications*, **42**, 259–268.
13. Rachev, S. T., Menn, C., Fabozzi, F. J. (2005). *Fat-Tailed and Skewed Asset Return Distributions: Implications for Risk Management, Portfolio Selection, and Option Pricing.* Wiley.
14. Rather, A.M., Agarwal, A., Sastry, V.N. (2015). Recurrent neural network and a hybrid model for prediction of stock returns. *Expert Systems with Applications*, **42**, 3234–3241.
15. Sermpinis, G., Theofilatos, K., Karathanasopoulos, A., Georgopoulos, E, Dunis, C. (2013). Forecasting foreign exchange rates with adaptive neural networks using radial-basis functions and Particle Swarm Optimization. *European Journal of Operational Research* **225**(3): 528–540.
16. Scott, D.W. (2015). *Multivariate Density Estimation: Theory, Practice, and Visualization.* John Wiley, New York.
17. Stoyanov, S., Rachev, S., Fabozzi, F. (2007). Optimal financial portfolios. *Applied Mathematical Finance* **14**(5): 401–436.
18. von Neumann, J., Morgenstern, O. (1944). *Theory of Games and Economic Behavior.* Princeton: Princeton University Press.

An Innovative Heuristic Mixed-Integer Optimization Approach for Multi-Criteria Optimization based Production Planning in the context of Production Smoothing

Felix Kamhuber[1*], Thomas Sobottka[1], Peter Schieder[1],
Maximilian Ulrich[1] and Wilfried Sihn[12]

1. Fraunhofer Austria Research GmbH, Theresianumgasse 27, Vienna 1040, Austria | Felix.Kamhuber@fraunhofer.at
Thomas.Sobottka@fraunhofer.at | Peter.Schieder@fraunhofer.at | Maximilian.Ulrich@fraunhofer.at

2. Vienna University of Technology (TU Wien), Institute of Management Science,
Theresianumgasse 27, Vienna 1040, Austria | Wilfried.Sihn@tuwien.ac.at

Abstract: This paper introduces an innovative heuristic mixed-integer optimization approach for multi-criteria optimization based production planning with rolling horizon for a discrete goods manufacturer. The fast moving consuming goods industry is characterized by standard and promotion sales volumes with different properties. State-of-the-Art multi-objective solution methods [1-3, 8-11] fail to address these properties adequately, due to the lack of considered subdimensions within a planning level. Furthermore these techniques contain static constraints rendering them unable to adapt the production system to seasonal (off-) peaks and to consider resource adjustments. In contrast, the presented approach features dynamic capacity-based restrictions and dynamic stock-levels within a given planning horizon. The product volumes per planning period (week) are split into two different subdimensions with specific constraints for order shifting and lot splitting for each subdimension and product. This approach pursues the optimized capacity utilization for a key production unit, featuring integer-based dynamic capacity restrictions. In addition to a smoothed production, mid-term stock-levels are simultaneously being optimized. The results show a 40% reduced output variation rate of the cost- and labor-intensive key equipment and a 30% reduced capacity requirement for downstream production equipment, compared with the initial manually compiled solution. Finally, the authors give an outlook on a possible enhancement of the method with statistical learning using periodic feedback from the production system.

Keywords: multi-product multi-period multi-objective production planning problem, heuristic optimization, lot splitting, production smoothing, rolling horizon, dynamic capacity constraints, dynamic stock-levels

1 Introduction

Modern production planning solutions are key enablers for today's highly competitive industrial environment. They (in-) directly influence the production system performance and profit of a factory. Different problem-specific heuristic [1, 5] and metaheuristic [2-3, 8-10] approaches characterized in [20] are proposed to meet the requirements of a *smoother* production. The latter ensures low flexibility costs and a good capacity utilization. An overview about practical and modelling issues as well as solution approaches concerning production smoothing problems presented in various contributions [1-5, 12, 19, 20] is given in section 3.

Trends in global and flexible markets, in combination with increasing product individualization, necessitate more flexible and responsive production systems [5, 7]. According to a Fraunhofer study [6], increasing variations in demand require sophisticated solutions concerning capacity flexibility measures, especially in the planning dimension week (47%).

This paper introduces an innovative heuristic optimization approach for a capacitated *Multi-Objective Multi-Product Multi-Period (MOMPMP)* production planning problem. The problem of scheduling the final production level is known as the *Production Smoothing problem* (PSP) [5].

This paper is structured as follows: After an introduction of the case study in section 2 and a literature review for the specific problem class at hand (see section 3), the developed heuristic optimization method is presented within section 4. A summary and discussion of the results in section 5 and an outlook on future research in section 6 conclude the paper.

[*] Corresponding author. Tel.: +43-676-888-61608; Fax: +43-1-504-6910-90. *E-mail address:* felix.kamhuber@fraunhofer.at

2 Case-study introduction and solution approach: An overview

This paper is centred on a case-study featuring an industrial European food producer aiming to introduce a smoother production flow to optimize the utilization of key production equipment/machines (see figure 1, production step 1). The key equipment represents a cost- and labor-intensive bottleneck unit within a soon-to-be-built factory. This production step influences other downstream production equipment within the total production system. The main task is the creation of an optimal, long-term (78 weeks in this use-case) dynamic capacitive-restricted production plan with a rolling horizon and the following optimization criteria:

goal (1) Reduced production volume peaks per planning period within the capacitive-restricted production system for all product types within a given planning horizon

goal (2) Optimized capacity utilization of a key production unit and downstream equipment

goal (3) Optimized stock levels for all product types and periods achieved and applied within a certain customer defined time frame

This specific optimization task should satisfy all relevant constraints concerning capacity, production process and product-type. In addition to reduced peaks in capacity demand through production smoothing, this heuristic optimization method enables a better capacity utilization resulting in a more energy-efficient production. It also prolongs the period until an investment in new capacity (i.e. machines) is necessary. Figure 1 provides a simplified process overview. An additional process step ('Extended maturing') was included as a buffer into the original process sequence to enable the operation of the production smoothing heuristic.

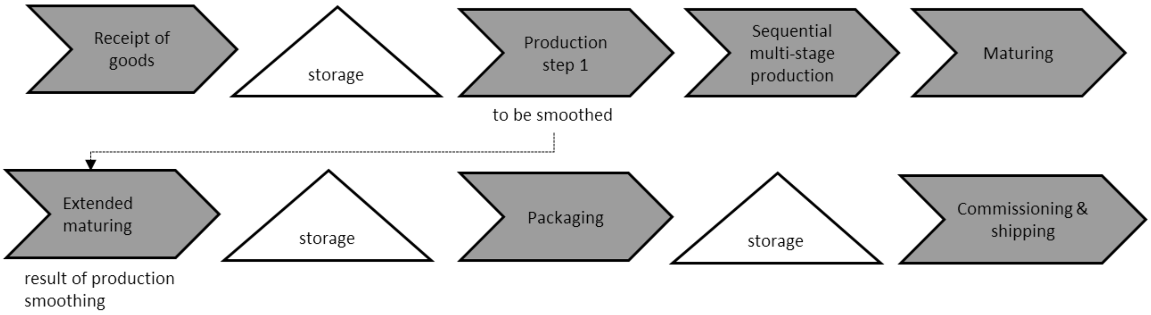


Figure 1: Production process system overview

The new factory has been planned using reliable forecasts for sales volume development to generate an initial production plan (→ initial solution). The heuristic has been originally applied with *goal (1)* and *goal* (2) to create the smoothed production volumes per period followed by a comprehensive discrete simulation study [17] to determine and validate the required capacities (→ production units) within the new production facility. Approaching the Go-Live of the plant in combination with the introduction of the rolling horizon the target function has been enhanced by *goal (3)* to further increase system productivity and process stability.

3 Literature review

Multi-objective multi-product multi-period (MOMPMP) production planning problems are NP-hard [1] and thus necessitate tailor-made, problem-specific (meta-) heuristics [20] to approximately solve problems of this complexity class within a reasonable time. The authors of [15, 19] give an overview of published literature dealing with MPMP models. Different multi-objective solution methods, including diverse mathematical models, are applied to deal with the complexity of purposeful search in the given solution space, namely:

- (MOGA) Multi-objective genetic algorithms [2, 8, 9, 12, 15]
- (Hybrid) Metaheuristics for (Mixed) Integer linear programming (MILP, ILP) models [3, 15, 16, 21]
- (MOPSO) Multi-objective particle swarm optimization [10], (MOBA) multi-obj. bat algorithm [18]
- Dynamic programming (DP) algorithms [5] and memory-based algorithms [11]
- Problem-specific heuristics [1, 5, 13] and rule-based algorithms [14]

In [5] the authors first present a dynamic programming algorithm for the exact solution of the corresponding production smoothing problem requiring significant computational effort, rendering its use impractical in a given real world environment. Thus, a 2-phase-metaheuristic approach is proposed – the problem is split into a constrained *batching problem* (BP) and *sequencing problem*. This approach offers some similarities with the proposed method in this paper, because jobs are split into smaller batches that can be consolidated with batches in other periods of the same product type. The main difference and shortcoming of existing (meta-) heuristics is the missing utilization of subdimensions within a certain planning dimension. Furthermore, the proposed rolling horizon approach considers dynamic capacity and dynamic target stock-level bounds, thus enabling the consideration of seasonal effects and resource (labor, production equipment) adjustments. The result evaluation is performed with an *index (1)* performing a weighted comparison of each part-goal result value.

4 Developed heuristic optimization method

The presented heuristic mixed-integer approach is modular and consists of five sequential optimization phases. The target function f(x) including its three defined part-goals calculates a weighted and normalized fitness-value, the « multi-criteria smoothing index » (MCSMI). This target function includes an evaluation function for a smoothed production planning horizon (f_1), next to an evaluation for optimized stock-levels (f_2) and an evaluation function to measure the smoothed capacity utilization (f_3):

$$\boldsymbol{Minimize}\, f(x) = \omega_1 \sum_{\substack{i=1\\j=1}}^{\substack{j=m\\i=n}} f_1(k\_prod_{ij}) + \omega_2 \sum_{\substack{i=1\\j=1}}^{\substack{j=m\\i=n}} f_2(k\_stock_{ij}) + \omega_3 \sum_{\substack{i=1\\c=1}}^{\substack{c=o\\i=n}} f_3(k\_capacity_{ic}) + \omega_4 \sum_{i=1}^{i=n} f_4(k\_plant_i)\,, \tag{1}$$

$$\boldsymbol{Subject\ to} \sum_{j=1}^{m} q_{ij} \le q_{max_i}, \forall i, \tag{2}$$

$$q_{ij} \in A_i, q_{max_i} \in A_i, \tag{3}$$

$$delta\_q_{prod_{ij}} \le\ delta_{max}\,, \forall delta_{q_{prod_{ij}}}, \tag{4}$$

$$delta_{Max} \in A_i \tag{5}$$

Legend:

ω_1 - ω_4 ... *part – goal weights*
m ... *total number of product types*
n ... *total number of periods within the planning horizon*
o ... *total number of capacity – units within the planning horizon*
f_1 ... *evaluation function for article production gradient (based on product type and planning period)*
f_2 ... *evaluation function for stock – level gradient (based on product type and planning period)*
f_3 ... *evaluation function for capacity – utilization gradient (based on capacity unit, planning period and amount of utilization)*
f_4 ... *evaluation function for plant production gradient (based on cumulated product type production gradient per planning period)*
k_prod_{ij} ... *production gradient for product type j in period i*
k_stock_{ij} ... *stock – level gradient for product type j in period i*
$k\_capacity_{ic}$ ... *capacity – level gradient for period i on capacity c*
k_plant_i ... *plant production gradient for cumulated sum of article production gradient in period i*
q_{ij} ... *production (share) quantity of product type j in period i*
slq_{ij} ... *stock-level quantity of product type j in period i*
$target_{slq_{ij}}$ ...*dynamic target stock-level quantity of product type j in period i*
cul_{ic} ... *capacity-unit level in period i on capacity c*
q_{max_i} ... *capacity constraint: max. allowed production quantity in period i*
$delta\_q_{prod_{ij}}$ ... *offset quantity of product type j in period i*
$delta_{max}$ ... *integer constraint for max. allowed offset [in periods]*
A_i ... *set of acceptable values for* $q_{ij}, q_{max_i}, delta\_q_{prod_{ij}}, delta_{Max}$

The defined part-goal functions $f_1 - f_4$ are represented by gradients provided in the following format:

$$f_1(k\_prod_{ij}) = Abs|q_{i+1,j} - q_{ij}| \tag{6}$$

$$f_2(k\_stock_{ij}) = Abs|slq_{ij} - target\_slq_{ij}| \tag{7}$$

$$f_3(k\_capacity_{ic}) = Abs|cul_{i+1,c} - cul_{ic}| \tag{8}$$

$$f_4(k\_plant_i) = \sum_{j=1}^{j=m} Abs|q_{i+1,j} - q_{ij}| \tag{9}$$

A gradient function, f_1 for instance, is calculated from (absolute) changes in production volumes between adjacent time periods. When compared with relative gradients, absolute gradients are preferred for evaluation because they penalize large deviations much stronger, and relative gradients could distract the algorithm from its goal. The decision maker's preferences are represented by the weights (equally set to 1 by default).

The given periodic-dynamic production capacity constraint (2) is a hard constraint and must strictly be fulfilled in each period to generate a valid convertible solution. Constraint (3) specifies that A_i, a set of specific multiples of certain integer values, according to an A/B/C categorization, is allowed for production only (see table 1, with « 1 » as the smallest possible batch unit of a product type).

Table 1 : Acceptable multiples for (q_{ij}, q_{max_i}) according their article specific ABC value

Parameters / Multiples of ABC value	A	B	C
q_{ij}, q_{max_i}	6	3 ‖ 6	1

Constraints (4) and (5) specify that the applied article-specific offsets for shifted production volumes must be lower than the maximum allowed offset values from the master data (→ technological requirement). There are two given maximum *integer* (→ planning periods) offset values per product type, one for standard sales volumes and one for promotion sales volumes. Table 2 lists the core functional requirements and constraints.

Table 2: Functional requirements for the optimization modules

Module, title and evaluation function	Description and constraints
Module 1: Production smoothing of promotion sales volumes (f_1)	This module reduces peak volumes of promotion-sales for each product and period according to flexible restrictions determined by product type, planning period, capacity and A/B/C (→ formulas (6) & (9)) category.
Module 2: Production smoothing of standard sales volumes (f_1)	This module reduces standard peak volumes for each product and period, according to flexible restrictions determined by product type, planning period, capacity and A/B/C (→ formulas (6) & (9)) category.
Module 3: Stock-Level smoothing (f_2)	This module optimizes defined mid-term stock-levels (see figure 3) in order to comply with target stock-levels according to formula (7).
Module 4: Capacity utilization forecast calculation (f_3)	This module calculates the capacity utilization forecast. This forecast is evaluated according to formula (8).

The multi-objective MPMP algorithm (see Algorithm 1), consists of five phases, which manipulate the production program towards different part-goals that are summed up in cost function $f(x)$.

Phase 1: Processing of master data, current production and demand plan (rolling horizon → each period)
Phase 2a: Minimization of promotion sales volumes peaks in the production plan. This phase of the algorithm calculates an average weekly production load (→ calculated from cumulated (→ C-articles) or article-specific (→ A/B articles) standard production volumes) and tries to shift shares of specific promotion volumes, which are above the average production load, into weeks that are below the average production load (gaps), as visualized in figure 2. These gaps are filled up according to the following heuristic:

(1) The gaps in periods closest to the current period are filled up first, until the current «*average production load*» *(APL)* per period is achieved (see Algorithm 1, rows 15 – 29)

(2) The remaining quantities are shared between all offset-periods, resulting in a higher *APL*

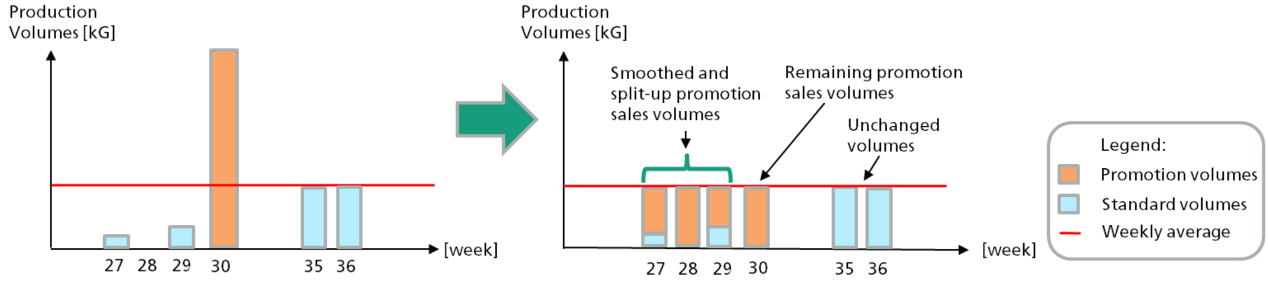


Figure 2: Promotion sales volumes smoothing (example)

The shifting of production volumes (promotion, standard) takes into account maximum offset intervals *(4).* In addition shifting is only possible into earlier production weeks (due to the structure of the input data) in order to still meet the production deadlines. Within the preparation phase, the input sales volumes have to be checked and accumulated to calculate product-specific gradients for production, capacity and stock-levels.

Phase 2b: Standard volume smoothing uses the same approach as phase 2a. An average production load (→ calculated from both standard and smoothed promotion volumes after phase 2a) is determined and standard production volumes that are above this derived average production load are preferably shifted to weeks that are below the considered average according to the heuristic in phase 2a.

Phase 2a and 2b feature some tweaks, e.g. a floating re-calculation of the average production load value. Another performance gain results from using smoothing indices for each phase, ensuring that an over-control is avoided by allowing a small percentage value to remain above the average production load value. This tweak further reduces the amount of production volume that needs to be shifted significantly. Optimization tests have shown this factor to be around 7-10%, depending on the specific dataset received.

Phase 3 - 4: Within these phases the filling level of each carrier-unit (rack) for the products, is evaluated and optimized according to its specific capacity utilisation and under the production restrictions (2) – (5). According to these restrictions, production volumes have to be a multiple of the minimum production volume of a product and are rounded tactically to half or full racks for each product in each period, according to the corresponding A/B/C-category of each product (see table 1). Depending on the stock level, rounding attempts to reach the optimal stock level, either by rounding up or down (or commercial) to increase, decrease or keep a certain stock level.

Phase 5: Smoothing via stock-level strategies is used (in addition to the stock level optimization of phase 3 - 4) to keep stock levels from a certain (shorter than in phase 3 - 4)) time frame close to the defined target stock level. This is done by calculating an average expected stock level over a certain number of weeks (adjustable, an example for this is « lead time 2 » in figure 3) and adding/removing the difference between the calculated stock level and the preferred stock level to/from the production plan (see the green dashed line in figure 3). Adding or removing production volumes is performed considering the average production volume while trying to maintain a smoothed production plan. Figure 3 shows a typical stock-level trend of a product (lead time = 12 weeks; given S_{min} , S_{max}). The goal is to improve the stock-level of the the influenceable time-period only, and not of the full planning horizon, since this would not be beneficial. This results in a short phase 5 with steep improvements of the objective function value realized within a longer time-span (see figures 3 – 4).

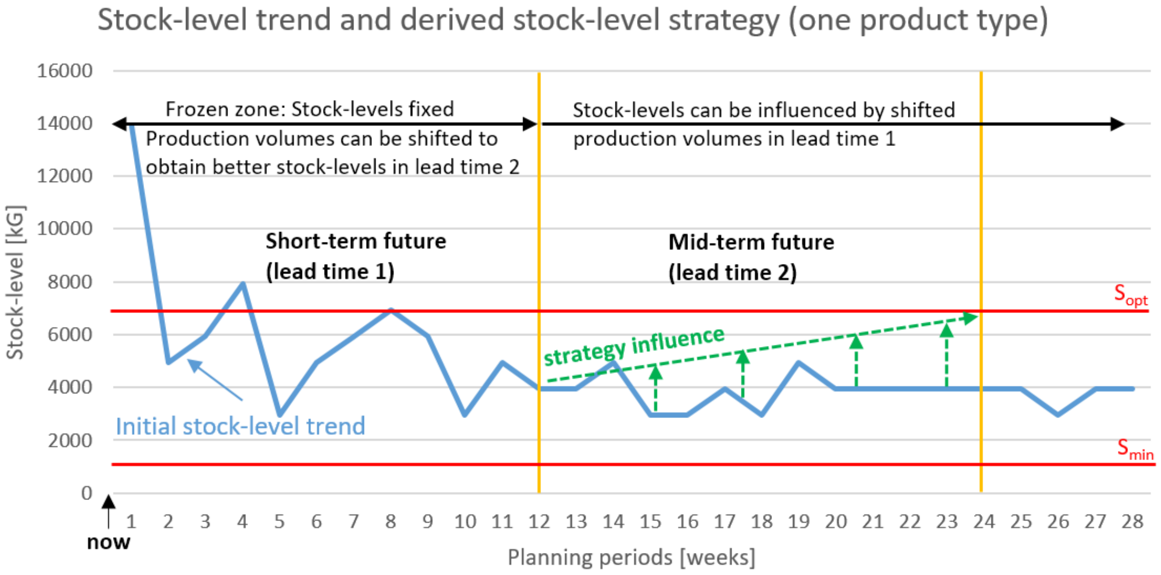


Figure 3: Stock-level strategies based on the specific stock-levels in the mid-term future

The final result is a multi-objective smoothed production plan for all products and the full planning horizon. It considers the planning dimensions «production volumes», split into the promotion and standard sales volumes dimension, next to the planning dimensions «stock-levels» and «capacity utilization».

Algorithm 1: The Multi-Objective MPMP Production Smoothing Heuristic

Phase 1: Preparation

```
Begin // process current master data, production and demand plan (rolling horizon → each period)
Calculate production plan solution matrix ppsm(m,n) // matrix with m (products) rows and n (periods) columns
Calculate stock-level gradient matrix slgm(m,n) // matrix with m rows and n columns
Calculate capacity utilization forecast matrix cufm(m,o) // matrix with m rows and o (capacity types) columns
$f(x) \longleftarrow$ Evaluate $(ppsm, slgm, cufm)$
$m' \longleftarrow m$ (derive prio-list $m'$) // priorisation by ABC analysis based on yearly average volumes and marginal returns
Update $ppsm' \longleftarrow ppsm$; $sglm' \longleftarrow sglm$ // update production and stock-level matrices
$(ppsm0'{}_{,} ppsm1') \longleftarrow split\ ppsm'$ // performs split into two matrices : for standard (0) and promotion (1) sales volumes
```

Phase 2a: promotion volume smoothing

```
For j = 1 to m
    For i = n to 1 // starting with the last period iterating to the first
        Calculate $mj \longleftarrow \mathrm{mean}(\mathrm{j}, \mathrm{n}/3)$ // floating mean value valid for product j for $\mathrm{n}/3$)periods
        If (i = n) Then $y \longleftarrow ppsm1'(i,j)$ // derive production promotion amount for product j in period $i$
        Else $y \longleftarrow ppsm1''(i,j)$
        End If
        $delta \longleftarrow compare\ (y - mj)$
        If (delta > 0) Then
            If ($j\ \epsilon(A||B)$) Then // if product type equals (A || B), stores week indices/quantities
                                        // in bwl HashMap, considering capacity and article constraints
                $bwl \longleftarrow$ search best local weeks including their quantities
                checkCapacityAndArticleConstraints $(ppsm', bwl)$
                Update $ppsm1'' \longleftarrow ppsm1'(bwl)$ // update ppsm1' by using bwl
            Else        // stores week indices/quantities in bwg HashMap incl. capacity/art. constraints
                bwg $\longleftarrow$ search best global weeks including their quantities
                checkCapacityAndArticleConstraints$(ppsm', bwg)$
                Update $ppsm1'' \longleftarrow ppsm1'(bwg)$ // update ppsm1' by using bwg
            End If
            Update $cufm' \longleftarrow cufm$ // capacity-utilization forecast update
            Update $ppsm'' \longleftarrow (ppsm0', ppsm1'')$; $f(x) \longleftarrow$ Evaluate $(ppsm'', slgm, cufm')$
        End If
    End For // next period for this article
End for // next article from prio-list $m'$
```

Phase 2b: standard volume smoothing

```
Repeat 2b $\longleftarrow$ (2a) : Update $ppsm0'' \longleftarrow ppsm0'$ // repeat steps of (2a) for (2b) with $ppsm0'$ instead of $ppsm1'$
    If (Update ( $ppsm0'' \longleftarrow ppsm0'(bwg\ ||bwl)$ Then
        Update $sglm'' \longleftarrow sglm'$ // actualise stock-level matrix (standard volumes!) each iteration
    // Update $ppsm''' \longleftarrow (ppsm0'', ppsm1'')$ instead of $ppsm'' \longleftarrow (ppsm0', ppsm1'')$;
// Update $cufm'' \longleftarrow cufm'$; instead of $cufm' \longleftarrow cufm$; → Evaluate $(ppsm''', slgm'', cufm'')$
    End if
// capacity utilization forecast finished → start rack-optimization
```

Phase 3 - 4 : rack optimization

```
For j = 1 to m
    For i = 1 to n
    Round $(ppsm''' \longleftarrow ppsm(i,j)''')$ // round tactically to half ||full racks for each product in
                                          // each period according to // A/B/C depending on stock levels
    // → according to target function part $f2(x)$ : roundings support to better achieve optimal stock-levels
    Update $(sglm''' \longleftarrow sglm(i,j)'')$; Update $(cufm''' \longleftarrow cufm'')$; $f(x) \longleftarrow$ Evaluate $(ppsm''')$
    End For
End For
// Phase 3 – 4 finished → perform mid-term stock-level strategies for all products and certain time frames
```

Phase 5 : stock-level strategies

```
For j = 1 to m
    Derive $optStockLevel\ (osl)$; $minStockLevel\ (msl)$; $leadTime\ (lT) \longleftarrow$ master data matrix (mdm)
    Initialize $delta_{stock} \longleftarrow 0$
    For i = $lT$ to $2 * lT$ // definition to perform changes in leadTime «1 » based on stockLevel in leadTime « 2 »
        Calculate/Update $delta_{stock} \longleftarrow$ Compare Abs$(osl - actualStockLevel(i,j))$
    End For
    If ($delta_{stock}$< msl) Then
        $bwl \longleftarrow$ search best local weeks including their quantities to fill 1st local / 2nd global stock-level gaps
        checkCapacityAndArticleConstraints($ppsm'''$, $bwl$)
    Else If ($delta_{stock} > osl * toleranceFactor$) Then
        $bwl \longleftarrow$ search best (a) local/ (b) global weeks incl. their quantities to downlevel current stock-level
    End If
    For i = 1 to $lT$ // perform production changes «fast » within 1st lead time
        While (bwl !isEmpty()) Do
            Update $ppsm'''' \longleftarrow ppsm0(\mathrm{i,j})'''(bwl)$ // update ppsm by using bwl on $ippsm0'''$
            Update $(sglm'''' \longleftarrow sglm(i,j)''')$ ; Update $(cufm'''' \longleftarrow cufm''')$ ;
             $f(x) \longleftarrow$ Evaluate $(ppsm'''')$
        End While
    End For
End For
Return $(ippsm''''; isglm''''; icufm'''')$ // print final optimized multi-criteria solution; End
```

5 Optimization results

The optimization results presented were obtained by a given real-life scenario with a specific dataset (78 planning weeks: 13/2018 – 38/2019) and the default settings of the algorithm.

The 2nd optimization phase, representing the core-unit (three of four part-goals are improved) of the proposed multi-phase heuristic approach, includes the production smoothing steps for promotion and standard volumes and accounts for 25% of global goal optimization. The third step is mandatory for the final rack preparation. However, it does not improve the global goal. The fourth optimization step raises or reduces stock-levels in order to create full racks and to simultaneously pursue the target stock level for each product in each period (fig. 5). Achieving filled racks maximizes the capacity utilization of the racks. The 5th optimization phase tries to shift production volumes in the near future to achieve optimal stock-values in the mid-term figure (fig. 2).

Figure 4 features the global goal optimization results (represented as trend), while figure 5 shows the individual part goal results. Figure 5 shows that the total capacity gradient (aggregated by 5 sub-capacity gradients itself) is affected by the optimization of the production gradient in phase 2. The production and capacity gradient optimization within phase 2 is derived at the cost of a worse stock-level gradient.

These results are achieved by shifting only ~ 8 – 10% of the annual production volume. However, the smoothing requires (1) lots to be split into part lots (→ shifting of part lots into existing other lots physically only leads in total to 5 – 7% more lots) and (2) additional (simple conditioning) storage capacity (fig. 1). These conditions result in additional – compared with the savings considerably lower – investment and setup costs.

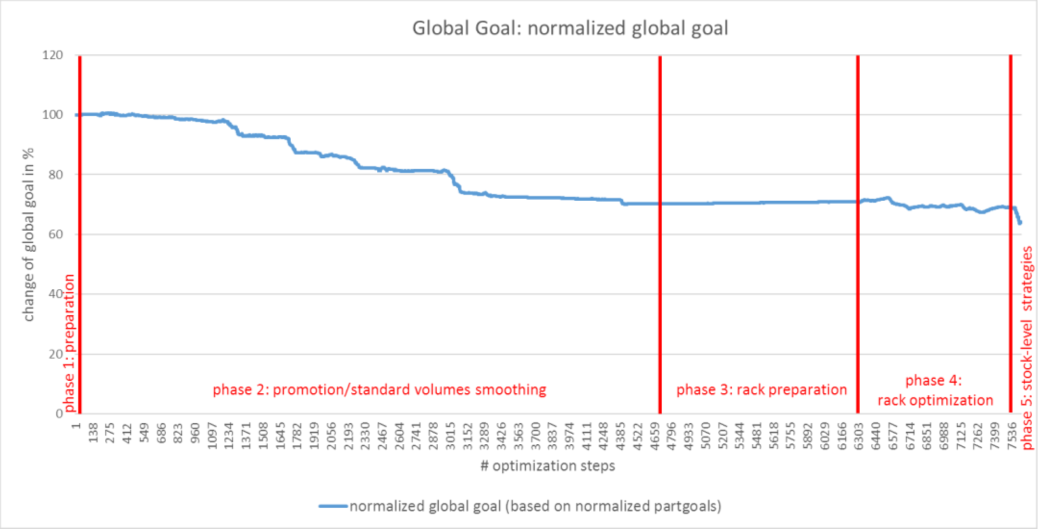


Figure 4: Global goal optimization

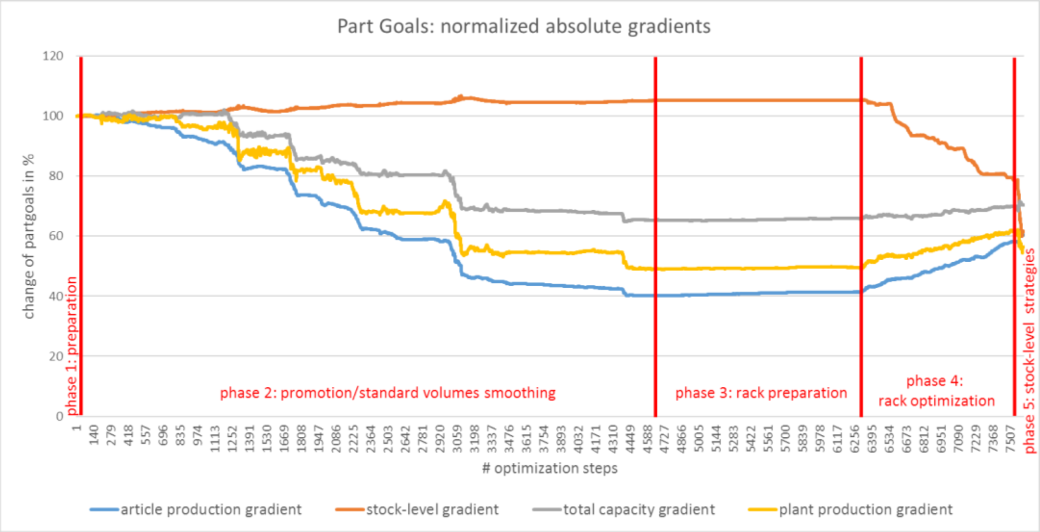


Figure 5: Individual part-goal optimization

6 Outlook and conclusion

The global goal optimization results release – compared with the initial production plan – a production smoothing potential of approximately 30%. Despite the fact that the absolute stock-level gradient cannot be measured explicitly in terms of minimized costs, the total cost saving potential from the production and capacity gradient is around 30%. This results in significantly lower investments for the new factory (fewer production units necessary) and 40% reduced operation costs on the key equipment (→ reduced labour costs achieved with balanced operation times) compared with – in comparison – very low additional costs for the additional process step. The actual optimization potential varies slightly with the specific planning scenario. Additionally, the method promises to be scalable as it becomes more effective with increasing appearance of peaks – and according to forecasts, more peaks are indeed to be expected in the future. Furthermore, this method enables an around 30% more uniform capacity utilization, resulting in a more energy-efficient production (because fewer units are in operation at a certain moment) and a longer lasting production facility configuration before the next expansion stage becomes necessary.

Future work on this method include a dedicated optimization of the capacity utilization – that is currently only implicitly optimized (see fig. 5), and an additional smoothing of packaging processes downstream of the hitherto considered production steps. Another important issue is – besides the comparison of this heuristic optimization method with a metaheuristic approach – to evaluate the influence of production as well as environmental conditions on the production plants' energy consumption. Therefore, after the collection of energy and production data, which is currently under way, the collected data will be analysed in order to find feasible opportunities - e.g. correlations between the production plan and the energy consumption. Thus, energy consumption will be integrated as an explicit optimization feature into the algorithm.

Another line of follow-up research is the algorithm-based analysis of collected production data, in order to better and automatically adjust the article specific constraints and stock-level strategies in the future. This learning from historic production data will lead to the algorithm becoming adaptive, thus unlocking further optimization potential currently inaccessible due to the fixed parameters, constraints and strategies.

7 Acknowledgements

This work is part of the research project ASPeCT, funded by the Austrian Research Promotion Agency (FFG) (Project number 858655). The authors would like to thank all project partners for their contributions.

References

[1] M. Karimi-Nasab, I. Konstantaras. A random search heuristic for a multi-objective production planning. *Computers & Industrial Engineering* (2012) 479--490.

[2] M. Karimi-Nasab, M. B. Aryanezhad. A multi-objective production smoothing model with compressible operating times. *Applied Mathematical Modelling* (2011) 35, 3596–3610.

[3] M. S. Al-Ashhab. An Optimization Model for Multi-period Multi-Product Multi-objective Production Planning, *International Journal of Engineering & Technology* (2016), IJET-IJENS Vol:16 No:01

[4] M. Yavuz, E. Akçali. Mesut Yavuz & Elif Akçali. Production smoothing in just-in-time manufacturing systems: a review of the models and solution approaches, *International Journal of Production Research* (2007), 45:16, 3579-3597.

[5] M. Yavuz, E. Akçali, S.Tüfekçi. Optimizing production smoothing decisions via batch selection for mixed-model just-in-time manufacturing systems with arbitrary setup and processing times, *International Journal of Production Research, 44*:15 (2006), 3061 – 3081.

[6] D. Spath, O. Ganschar, S. Gerlach, M. Hämmerle, T. Krause, S. Schlund. Produktionsarbeit der Zukunft - Industrie 4.0. Fraunhofer-Institut für Arbeitswirtschaft und Organisation IAO, Stuttgart (2013), ISBN 978-3-8396-0570-7.

[7] C. Morawetz. Vorgehensmodell zur Entwicklung eines Entscheidungsunterstützungssystems zur kostenoptimalen mittelfristigen Kapazitätsanpassung, doctoral thesis, Vienna University of Technology (2015).

[8] M. M. Soares, G.E. Vieira. A new multi-objective optimization method for master production scheduling problems based on genetic algorithm, Int. J. Adv. Manuf. Technol. 41 (2009), 549-567.

[9] N. Chakraborti, B.S. Kumar, V.S. Babu, S. Moitra, A. Mukhopadhyay. A new multi-objective genetic algorithm applied to hot-rolling process, Appl. Math. Model. 32 (2008) 1781 – 1789.

[10] D. Y. Sha, H.H. Lin. A multi-objective PSO for job-shop scheduling problems, Expert Systems with Applications, 37:2 (2010) 1065 – 1070.

[11] A. Cheriet, F.Cherif. Solving dynamic multi-objective problems using a copula-based estimation of distribution algorithm, *Proceedings of 6th META* (2016) 78 – 80.

[12] O. Masmoudi, A.Yalaoui, Y.Ouazene, H.Chehade. Simultaneous lot-sizing and scheduling in a flow-shop system with energy consideration, *Proceedings of 6th META* (2016) 349 – 351.

[13] M. Karimi-Nasab, S.M.T. Fatemi Ghomi. Multi-objective production scheduling with controllable processing times and sequence-dependent setups for deteriorating items, *International Journal of Production Research*, 50:24 (2012), 7378 – 7400.

[14] Y. Monden. Toyota production system: An Integrated Approach to Just-In-Time. Taylor Francis Group, 4th edition (2012), ISBN 978-1-4398-2097-1.

[15] I. Saracoglu, S. Topaloglu, T. Keskinturk. A genetic algorithm approach for multi-product multi-period continuous review inventory models, Expert Systems with Applications 41 (2014), 8189 – 8202

[16] S.Torkaman, S.M.T Fatemi Ghomi, B.Karimi. Multi-stage multi-product multi-period production planning with sequence dependent setups in closed-loop supply chain, Computers & Industrial Engineering 113 (2017) 602-613.

[17] T. Sobottka, F. Kamhuber, J. Henjes, W. Sihn. A case study for simulation and optimization based planning of production and logistics systems, Proceedings of the 2017 Winter Simulation Conference.

[18] A.E. Samrout, O. Braydi, R. Younes, F. Trouchou, P. Lafon. A new hybrid method to solve the multi-objective optimization problem for a composite hat-stiffened panel, *Proceedings of 6th META* (2016), 116 – 124.

[19] Z. Sazvar, S.M.J. Mirzapour Al-e-hashem, K. Govindan, B. Bahli. A novel mathematical model for a multi-period, multi-product optimal ordering problem considering expiry dates in a FEFO system, Transportation Research Part E: Logistics and Transportation Review, Volume 93 (2016), 232- 261.

[20] F. Ansari, K. Schenkelberg, U. Seidenberg & M. Fathi. Problem Solving in the Digital World: Synoptic Formalism, Incrementalism and Heuristics, Encyclopedia of Computer Science and Technology, 2nd Edition; Laplante, P., Ed.; Taylor & Francis: New York (2017).

[21] Talbi, EG. Annals of Operations Research (2016) Volume 240, 171-215, https://doi.org/10.1007/s10479-015-2034-y

New Approach for Continuous and Discrete Optimization: Optimization by Morphological Filters

Khelifa Chahinez Nour El houda[1] and Belmadani Abderrahim[2]
Laboratoire d'Optimisation des Réseaux Electriques LORE
Department of computer science

University of Science and Technology of Oran- Mohamed Boudiaf. Oran, Algeria

[1]*khelifa.chahinez@gmail.com, chahinez.khelifa@univ-usto.dz*
[2]*abderrahim.belmadani@gmail.com, abderrahim.belmadani@univ-usto.dz*

Abstract*:In this paper, we propose a new optimization algorithm inspired by image processing methods. Indeed, our algorithm of Optimization by Morphological Filters (OMF) uses the morphological transformations, more precisely the erosion, for the search of the global optimum in a multidimensional space.
We tested our OMF algorithm on a set of benchmark functions. The results obtained are very convincing, which shows that OMF is a very competitive algorithm for the resolution of optimization problems.*

Keywords*:* optimization, stochastic search, metaheuristics, numerical erosion, mathematical morphology.

1. Introduction

Metaheuristic algorithms are generally stochastic search with iterative behavior for solving complicated optimization problems.Many of them imitates natural phenomena such as Genetic Algorithm [1] which is inspired from evolutionary process. Others are based on swarm behavior such as Ant Colony [2],Bee Colony [3],ParticleSwarm Optimization [4],Cuckoo Search [5], Bat-inspired Algorithm [6]and Grey Wolf Optimizer [7]. Another metaheuristic that imitates the physical annealing process isSimulated Annealing[8], while Harmony Search Algorithm conceptualizes the musical process of searching for a perfect state of harmony[9].

In this paper,we describe a brief overview of existing metaheuristic algorithms. We outline the proposed OMF algorithm and to demonstrate the effectiveness and robustness of the OMF algorithmwe present the results and discussions of the application of OMF on benchmark functions from literature.

2. A brief overview of existing metaheuristic algorithm

Metaheuristics are often employed to solve hard optimization problemdue totheir simplicity, they are mostly inspired by very simple concepts. They are classifiedinto three main classes: Evolutionary algorithms (EA), inspired by the concept of evolution in nature, among them the Genetic Algorithm (GA) proposed by Holland on 1975 and developed by De jong and Golberg.GA simulates Darwinian evolution.Deferential Evolution[10], Evolution Strategy (ES) [11] and Evolutionary Programming[12] are also a type of EA. The second main branch of metaheuristics is swarm intelligence (SI). These algorithms imitate the social intelligence of creatures in swarms, flocks and herds. The particle swarm optimizer (PSO) algorithm proposed by Kennedy and Eberhant is the most popular algorithm of this branch; it is inspired by social behavior of birds flocking or fish schooling. Cuckoo Search algorithm,Grey Wolf Optimizer and Ant Colony optimizer are also a kind of SI algorithms. The third subclass of metaheuristics is physics-based algorithm such as Gravitational SearchAlgorithm [13]which emulates newton's law of universal gravitation, Big-Bang Big-Crunch (BBBC) [14], Charged System Search (CSS) [15], Central Force Optimization (CFO) [16], Artificial Chemical Reaction Optimization Algorithm (ACROA) [17], Black Hole (BH) [18], Curved Space Optimization[19]…

Besidethese three mains classes, there are some algorithms inspired from different phenomena such asHarmony Search optimization and its derivative forms(Global Harmony Search Algorithm GHS[20],Improved Harmony Search Algorithm IHS[21]). They are music-based metaheuristic algorithms inspired by the observation that the aim of music is to search for a perfect state of harmony.

3. Optimization by morphological filter:

In this paper, we propose a new stochastic optimization algorithm inspired by morphological transformations.OFM algorithm uses the erosionprocess which consists to find the minimum combination of pixel values in the neighborhood of morphological filter(structuring element in image processing).

Inspiration:

Mathematical morphology wasdeveloped from set theory. It was introduced by G. Matheron [22] in order to analyze geometric structure of metallic samples, than J.Serraextended it to image processing[23].It was initially developed to process binary images(mathematical morphology set) than extended to grayscale images. In this last case, mathematical morphology became functional mathematical morphology which is the object of our inspiration.

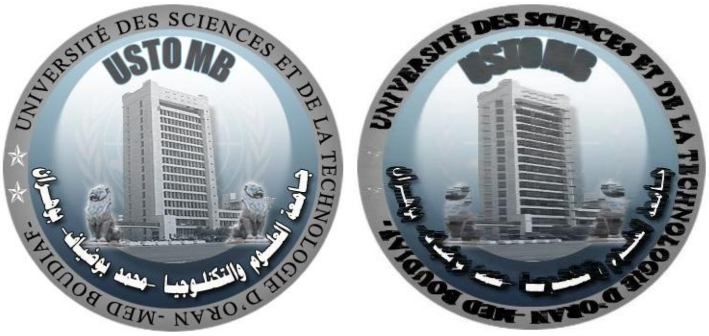
The basic idea of the mathematical morphology is to compare the analyzed image with structuring element (a set or function describing some shape namely Morphological Filter),its application allowsto obtain many information from the original image.The two fundamental operations of mathematical morphology are dilation and erosion; they are employed in order to enlarge (dilate) or reduce (erode) the size of features in images. They also may be combined to produce opening (erosion operation followed by a dilation) and closing (dilation operation followed by an erosion). The main morphological operation that inspired our algorithm is the functional erosion.

The functional erosion:

The functional erosion is applied to grayscale (multilevel) images in order to erode/shrink/reduce the regionsboundaries of foreground pixels. It corresponds to find minimum of pixel combinations and the Kernel function (structuring element), it can be defined as:

$\varepsilon B(f) = f \ominus B^t(x) = inf\{f(y), y \in B_X\}$*(1)*

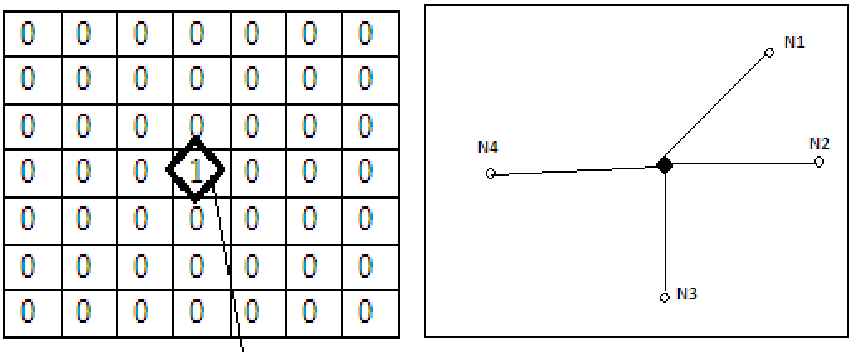
This transformation has the properties to reduce the “peaks” of gray levels and widen “the valley”;it tends to homogenize the image to darken it and to spread the edges of darkest objects. The figure1 below shows the result of erosion transformation by a flat structuring element(7*7).



(a) initial image (b)eroded image

Figure 1.Results of Erosion with a flat structuring element

To adapt the functional erosion to the proposed approach, the structuring element in erosion is replaced by a star where the center (C1: diamond) is the actual solution and N1, N2, N3 and N4 are randomly generated neighbors (morphological filter).



0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

Origin of structuring element

Figure 2.Structuring element Vs OFM filter

As previously mentioned, OFM is a stochastic approach which starts with random generation, all variables generated must be ranged in the interval [0, R] in order to have the hypercube with length R(search space). This procedure is assured by normalization process[24] defined as follow:

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} * R \qquad (2)$$

The algorithm usesmany filters operating in parallel.A filter applied to the objective function solution explores the neighborhood of its center and return the neighbor with the best fitness. The number of filtersand the number of neighbors are fixed at the initialization step. During the search process, if a better neighbor is found, the centre of the structuring element (filter) is moved on it.Otherwise, the size of the filter is reduced in order to check a closer neighborhood.This procedure is executed by all filters and our stop criterion is the exhaustion ofneighborhood search. The algorithm stops when the sizes of all the filters become too small (less than ε). The OMF steps can be summarized as follow:

```
Initialize: Min, Max, filter size, search space size;
Begin
REPEAT
For each filter do
Neighborhood calculation Procedure ();
Movement Procedure ();
Until the size of all filters ≤ ε;
Return the best solution found
End.
```

Figure 3.Pseudo code of the OMF algorithm

Where Neighborhood calculationand Movement Procedures are detailing the neighborhood calculation and movement applied to each filter, they are described down below:

```
 Neighborhood calculation Procedure ();
Begin
For all neighbors of all filter_center (X) do
$X`_{j,f} \overset{randomly}{\underline{\underline{\textbf{Or}}}} \begin{cases} X_{j,f}+a*filter\_size \\ X_{j,f}=random*R; \end{cases}$
End.
```

Figure 4.Pseudo code of the Neighborhood calculation Procedure

The *j-th* coordinate of each neighbor is calculated by taking one of the following options:do not change, shift left, shift right or take a random position. The first three possibilities are realized with the following equation:

$$X`_{j,f} = X_{j,f}+a*filter\_size(3)$$

Where filter_size denotes the size of the *f-th*filter and *a* is a parameter randomly chosen from the set {1, -1,0}.

To explore unvisited regions of the search space and to diversify the search, the *j-th*coordinate can be calculated using to the following formula:

$$X_{j,f}=random*R \qquad (4)$$

```
Movement Procedure ();
Begin
For all filters do
If any neighbor is better than the filter_centerthen Move to bestone
Else reduce the size of the filter;
Endif,
End.
```

Figure 5.Pseudo code of Movement Procedure

The figure 5 shows the movement procedure which explains the displacementin the filter. The neighbor which presentsthe best fitness becomes the new center of the actual filter.Otherwise the size of this filter will be reduced using equation (6).It is necessary to note that the initial size of all the filters can be taken equal to *R*.

$$Filter\_size = R/c^{k} \qquad (6)$$

Where:

R is therange of search space,K denotes the number of reductions of the size of the actual filter andC is a constant which is fixed in our case at *1.001*.

Figures 6-8 illustrate the neighborhood calculation and movement procedures with fitness function f(x, y) = x^2+y^2. In iteration 1, the center C1 has fitness=2 and neighbors coordinates are calculated using neighborhood calculation procedure (figure 4) ,where N1(-5,0), N2(0,1.25),N3(5,2.25), N4(0,-5) with f(N1)=25, f(N2)=1.56, f(N3)=6.25, f(N4)=25. The movement procedure, shown in figure 5, compares the fitness’s of the actual solution (C1) and its neighbors and performs a movement or a reduction of the parameter *filter_size*. Since f(N2) in iteration 1 is better than f(C1),N2 becomes the new center of the filter in iteration 2. contrariwise to iteration 1, no neighbor presents a better solution in iteration 2 so the algorithm will perform a reduction of the size of the filter. We note thatneighbors’coordinates are always generated by the neighborhood calculation procedure.

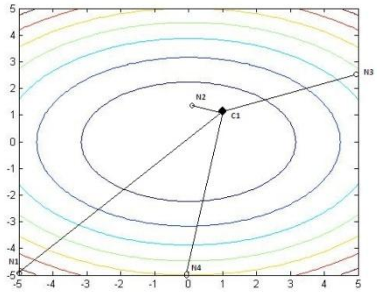


Figure 6.Iteration 1

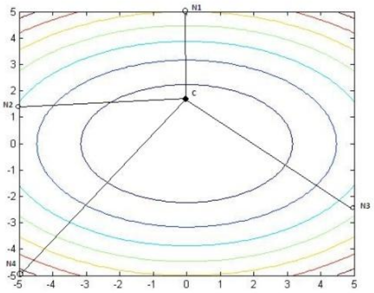


Figure 7.Iteration 2

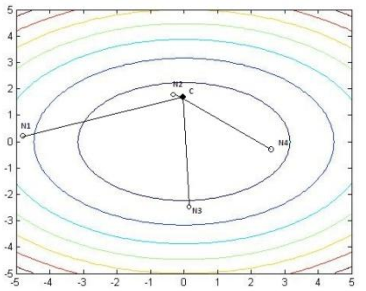
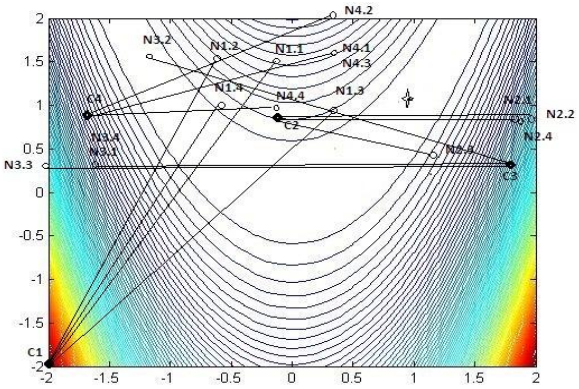


Figure 8.Iteration 3

In order to explain our approach, we choose to show the OMF behavior when the objective function is a two dimensional Rosenbrock function in the range [-2, 2]. For this example, we apply OMF using three filters with four neighbors and an initial *filter_size*=10.The stop criterion is epsilon=10^{-5}. The centers of the filters (Solutions) C1, C2 and C3 are represented by small black rhombus and the neighbors are represented by circles. The four neighbors of the *f-th* solution are Nf.1 to Nf.4.

Figures 9 and 10 show the solutions connected with their neighbors by a straight line. Figures 11-14 show only the position of the new solutions (centers of the filters) at the actual iteration.

The global optimum is marked at (x_1=1, x_2=1) where the function returns 0.



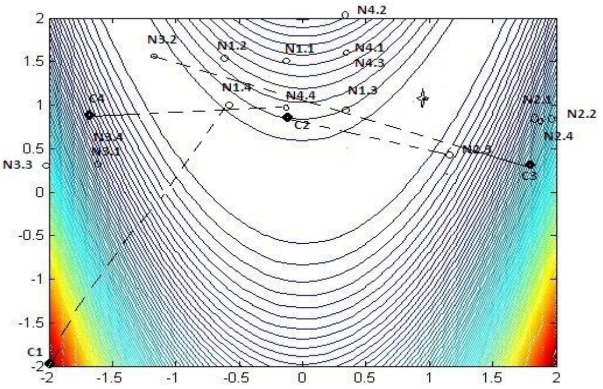


Figure 9.Iteration 1 **Figure 10.Movement**

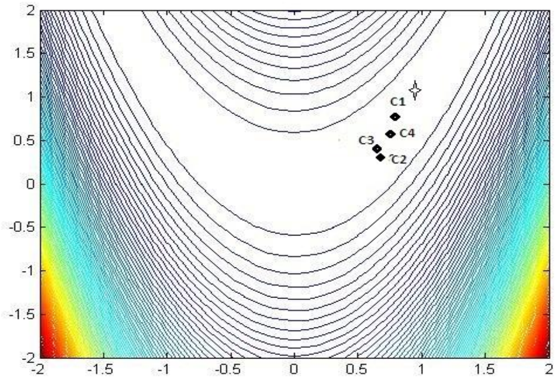


Figure 11.Iteration4722(25%)

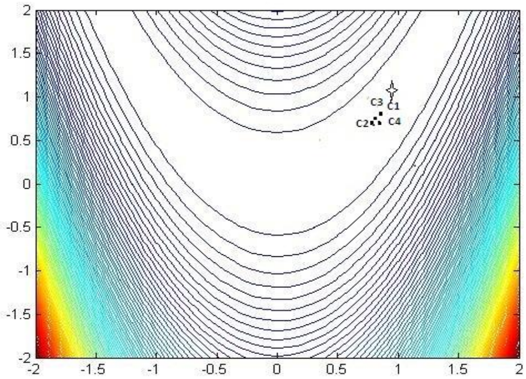


Figure 12.Iteration9445(50%)

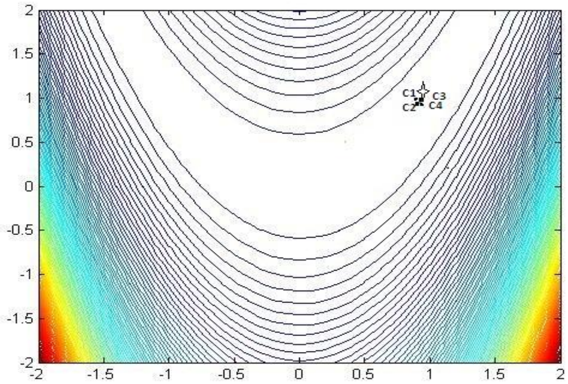


Figure 13.Iteration 14167(75%)

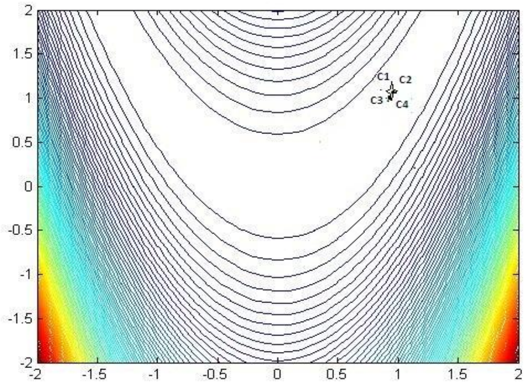


Figure 14. Iteration 18889(100%)

Figure 9 shows the results of the first iteration. Here, the positions of the centers of the filters are generated randomly and the calculation of the neighbors is performed using Neighborhood calculation Procedure.For example, the first coordinate of N1.1 is calculated by expression1 (equation 4) with a=0 (randomly chosen)while the first coordinate of N2.4 in iteration 2 is calculated with expression 2 (equation 5).

Iteration1:

- The first filter **C1** (-2,-2) with **fitness**=90.04482,its neighbors are:

  N1.1 (-0.10835,1.52559) with **fitness**=230.40353,

  N1.2 (-0.67978, 1.52559) with **fitness**=115.92217,

  N1.3 (0.46307, 0.95416)with **fitness**=55.00818,

  N1.4 (-0.67978, 0.95416) with **fitness**=27.03384;

- The second filter center **C2** (-0.10835, 0.95416)with **fitness**=1229.06620, its neighbors are:

  N2.1 (1.96606, 0.932362) with **fitness**=861.20809,

  N2.2 (2, 0.932362) with **fitness**=942.04000,

  N2.3 (1.39463, 0.36093) with **fitness**=251.08458,

  N2.4 (1.96606, 0.93236) with **fitness**=861.20809;

- The third filter center **C3** (1.96606, 0.36093) with **fitness**=259.53315, its neighbors are:
  N3.1(-1.59565, 0.38471) with **fitness**=473.89537,
  N3.2(-1.02422, 1.52757) with **fitness**=26.99790,
  N3.3(-2, 0. 38471)with **fitness**=1316.02620,
  N3.4(-1.59565, 0.38471) with **fitness**=473.89537;

- The fourth filter center **C4**(-1.59565, 0.95614) with **fitness**=229.82200, its neighbors are:
  N4.1(0.37332, 1.55049) with **fitness**=199.51868
  N4.2(0.37332,2) with **fitness**=346.58657,
  N4.3(0.37332, 1.55049) with **fitness**=199.51868,
  N4.4(-0.19810, 0.97906) with **fitness**=89.76120.

The movement procedure, shown in figure 5, compares the fitness's of each solution and its neighbors and performs a movement or a reduction of the parameter *filter_size*. Here, N1.4 becomes the new C1, N2.3 becomes the new C2, N3.2 becomes the new C3 and N4.4 becomes the new C4. In this example and at this iteration all the centers are moved however for the next iterations if no neighbor is performing better than the center than this one remain unchanged and the parameter *filter_size*is reduced using equation 6.

The results after 25%, 50%, 75% and 100% of the iterations are shown in Figure 11, figure 12, figure 13 and figure 14 respectively. We can see clearly the movement of all the centers toward the global optimum.

The final results of the algorithm are:

C1(0.999999429112051,0.999998278876708) and **fitness**=4.07653E-10;

C2(1.00000200647046, 1.00000602199272) and **fitness**=1.58660E-10;

C3 (0.999998892129236, 0.999999038982221) and **fitness**=1.10218E-10;

C4(0.999998988222136,0.999996931480444) and **fitness**=3.38902E-11.

4. Results and discussion:

In this section,OMF algorithm is tested using 13benchmark functions classically used by many researchers. The results are compared with those of other optimization algorithms. Thebenchmark functions are listed in table 1 where Fct indicates nouns of used functions, D indicates dimension of the function, Range Space (RS) is the boundary of the function's search space, and F_{min} is the theoretical global optimum.

TABLE I. BENCHMARK FUNCTIONS

Fct	Formulation	D	RS	F_{min}
Sphere	$f(x) = {}_{i=1}^{N} x_i^2$	30	[-100,100]	0
Schwefel problem	$\mathbf{f(x)} = {}_{i=1}^{N} \lvert \mathbf{x_i} \rvert + \prod_{i=1}^{N} \lvert \mathbf{x_i} \rvert$	30	[-10,10]	0
Rotated hyper ellipsoid	$\mathbf{f(x)} = \sum_{i=1}^{N} (\sum_{j=1}^{i} \mathbf{x_j})^2$	30	[-100,100]	0
Rosenbrock	$f(x) = {}_{i=1}^{n}(100 * (x_i - x_{i-1}^2)^2 + (x_{i-1} - 1)^2)$	30	[-30,30]	0
Step	$\mathbf{f(x)} = \sum_{i=1}^{N} (\mathbf{x_i} + \mathbf{0,5})^2$	30	[-100,100]	0
Schwefel function	$\mathbf{f(x)} = \sum_{i=1}^{N} (\mathbf{x_i} \sin(\sqrt{\lvert \mathbf{x_i} \rvert}))$	30	[-500,500]	**-12569.5**
Rastrigin	$\mathbf{f(x)} = {}_{i=1}^{N} (\mathbf{x_i^2} - \mathbf{10} \cos(\mathbf{2\pi x_i}) + \mathbf{10})$	30	[-5.12,5.12]	0
Ackley	$f(x) = -20 \exp\left(-0.2 \sqrt{\frac{1}{30} {}_{i=1}^{N} x_i^2} \right. - \exp(\frac{1}{30} \cos(2\pi x_i) + 20 + e^1)$	30	[-32,32]	0

Griewangk	$\mathrm{Df(x)} = \frac{1}{4000}\ {}_{i=1}^{N} x_i^2 - \prod_{i=1}^{N} \cos(\frac{x_i}{\sqrt{i}}) + 1$	30	[-600,600]	0
Michalewicz	$\mathbf{f(x) = -\sum_{i=1}^{N} \sin(x_i) * \sin\frac{i * x_i^{2}}{\pi}^{20}}$	30	[0,π]	-4.687
Six-Hump Camel	$\mathrm{F\ f(x)} = 4x^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	[-5,5]	-1.0316
Braninrcos	$f(x)=\left(X_2 - \frac{5.1}{4\pi^2}X_1^2 + \frac{5}{\pi}X_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos X_1 + 10$	2	[-5,5]	0.398
Golden-Price	$f(x) = \big(1 + (x_1 + x_2 + 1)^2 * (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x\ {}_1x_2 + 3x_2^2)\big) * (30 + (2x_1 - 3x_2)^2 * (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2))$	2	[-2,2]	3

The OMF algorithm was run 30 times on each benchmark function with number of filters *NB-filters*= 30, number of neighbors=10 and a minimum size of the filters (stop criterion: Size of each filter less than ε= 10^{-24})

The statistical results (average and standard deviation) are reported in Table 2.The results of OMFare compared to those of PSO[25]as an SI-based technique, GSA [13] as a physics-based algorithmin addition toGWOresults [7]and GHS results [20].

TABLE II. RESULTSFOR BENCHMARK FUNCTIONS OPTIMIZATION

Fct	**GWO**	**PSO**	**GSA**	**GHS**	**OMF**
Sphere	6.59E-28 (6.34E-5)	0.000136 (0.000202)	2.53E-16 (9.6E-17)	1.0 E-5 (2.2 E-5)	**3,32E-33 (2,7E-48)**
Schwefel problem	**7.18E-17 (0.029014**)	0.042144 (0.04542)	0.055655 (0.194074)	0.0728 (0.1144)	0,02058 (0)
Rotated hyper ellipsoid	**3.29E-06 (79.14958)**	70.12562 (22.11924)	896.5347 (318.9559)	5146.2 (6348.7)	0,050921 (7,05E-17)
Rosenbrock	26.81258 (69.90499)	96.71832 (60.11559)	67.54309 (62.22534)	49.669 (6348.8)	**0,145492 (5,64E-17)**
Step	0.816579 (0.000126)	0.000102 (8.28E-05)	2.5E-16 (1.74E-16)	**0 (0)**	**3,755E-33 (4,701E-33)**
Six-Hump Camel	-1.0316/ (-1.03163)	-1.03163 (6.25E-16)	-1.03163 (4.88E-16)	-1.0316 (0.000018)	**-1,0316 (0)**
Braninrcos	0.397889 (0.397887)	0.397887 (0)	0.397887 (0)	*******	**0,398 (0)**
Golden-Price	3.000028 (3)	3 (1.33E-15)	3 (4.17E-15)	*******	**3 (0)**
Schwefel function	-6123.1 (-4087.44)	-4841.29 (1152.814)	-2821.07 (493.0375)	-12569.46 (0.050)	**-12569,5 (1,85E-12)**
Rastrigin	0.310521 (47.35612)	46.70423 (11.62938)	25.96841 (7.470068)	**0.0086 (29)**	0,031064 (5,646E-17)

Ackley	**1.06E-13** **(0.077835)**	0.276015 (0.50901)	0.062087 (0.23628)	0.0209 (0.0216)	2,378E-7 (1,129E-16)
Griewangk	0.004485 (0.006659)	0.009215 (0.007724)	27.70154 (5.040343)	0.1024 (0.1756)	**0,00021** **(0)**
Michalewicz	4.042493 (4.252799)	3.627168 (2.560828)	5.859838 (3.831299)	NA	**-4,687** **(1,80E-15)**

According to Table II, OMF is able to provide very competitive results. OMF algorithm outperforms all others in optimization of Sphere function, Rosenbrock, Griewangk, Six-Hump Camel back, Braninrcos, Golden-Price, Schwefel function and Michalewicz function.

For many functions (Six-Hump Camel back, Braninrcos, Golden-Price and Griewangk)OMFis able to return exactly the global optimum.

Scalability study:

When the dimension of the functions increases from 30 to 100, the performance of the different methods degraded. As shown in TableIII, The comparison results of the OMF with HS and its variants show that the OMF is performing better in 4 cases.

TABLE III. RESULTS OF BENCHMARK FUNCTIONS DIM=100

Fct	HS	IHS	GHS	OMF
De Jong	8.683062 (0.775134)	8.840449 (0.762496)	**2.230721** **(0.565271)**	3449,978461 (1,3875E-12)
Schwefel's Problem	82.926284 (6.717904)	82.548978 (6.341707)	**19.020813** **(5.093733)**	43,136096 (1,445E-14)
Rosenbrock	16675172.1 (3182464.488466)	17277654.0 (2945544.275052)	2598652.617 (915937.797217)	**524266,4048** **(6,36811E-10)**
Rotated hyper-ellipsoid	215052.904398 (28276.375538)	213812.584732 (28305.249583)	321780.353575 (39589.041160)	**1,49E+05** **(2,96014E-11)**
Schwefel's function	-33937.364 (572.3904)	**-33596.899** **(731.1918)**	-40627.3455 (395.457330)	-36851,61011 (2,2214E-11)
Rastrigin	343.49779 (27.24538)	343.23204 (25.14946)	**80.657677** **(30.368471)**	192,0721007 (1,1563E-13)
Ackley's Path	13.857189 (0.284945)	13.801383 (0.530388)	8.767846 (0.880066)	**8,75E+00** **(3,61345E-15)**
Griewank	195.59257 (24.80835)	204.29151 (19.15717)	54.252289 (18.600195)	**32,04980615** **2,89076E-14**

Effect of stop criterion

The stop criterion is the smallest valueof the *filter_size*parameter; translated hereby ε. Table IV shows the effect of this parameteron the results of OMF algorithm.The algorithm is applied for the optimization of the benchmark functions of Table I with four different values of ε. We go from a larger size ε=10^{-9} to the smallest one ε=10^{-24}. We note That OMF gives the best results for the greater precision. That can be explained by the fact that the small precision makes the sweeping of the regions not visited before possible. However, some optimum found don't change significantly which shows a stability of the optimization process of OMF.

TABLE IV. THE EFFECT OF STOP CRITERION

Fct	**$\varepsilon=10^{-9}$**	**$\varepsilon=10^{-12}$**	**$\varepsilon=10^{-16}$**	**$\varepsilon=10^{-24}$**
Sphere	2,26E-16 (1,5E-31)	1,07E-22 (7,1E-38)	4,70E-31 (1,7E-46)	**3,32E-33** **(2,7E-48)**
Schwefel problem	0,09435 (0)	0,090409 (2,8E-17)	0,046469 (1,4E-17)	**0,02058** **(0)**
Rotated hyper ellipsoid	1,22E-01 (7,0E-17)	0,114811 (4,2E-17)	0,114811 (2,8E-17)	**0,050921** **(7,05E-17)**
Rosen brock	0,145492 (5,64E-17)	0,145492 (2,82144E-17)	0,145424 (0)	**0,145492** **(5,64E-17)**
Step	1,15E-17 (4,701E-33)	5,72E-23 (1,19493E-38)	5,36E-31 (2,67235E-46)	**3,755E-33** **(4,701E-33)**
Schwefel function	-12569,4 (1,8E-12)	-12569,4 (1,8E-12)	-12569,4 (1,8E-12)	**-12569,5** **(1,85E-12)**
Rastrigin	0,100070 (5,6E-17)	0,098044 (2,8E-17)	0,096518 (1,4E-17)	**0,031064** **(5,646E-17)**
Ackley	4,25E-01 (1,1E-16)	0,192977 (5,6E-17)	0,102348 (4,2E-17)	**2,378E-7** **(1,129E-16)**
Griewangk	0,00021 (0)	0,000219 (1,1E-19)	0,000219 (1,6E-19)	**0,00021** **(0)**
Michalewicz	-4,68765 (1,8E-15)	-4,68765 (1,8E-15)	-4,68765 (1,8E-15)	**-4,687** **(1,80E-15)**
Six-Hump Camel	-1,0316 (0)	-1,03164 (0)	-1,03162 (0)	**-1,0316** **(0)**
Braninrcos	0,397887 (0)	0,397887 (0)	0,397887 (0)	**0,398** **(0)**
Golden-Price	3 (0)	3 (0)	3 (0)	**3** **(0)**

Integer programming problem

Many real-world applications require the variables to be integers. These problems are called Integer Programming problems. We choose six common integer programming benchmark problems (see Table V) to investigate the performance of OMF.

TABLE V. INTEGER PROBLEM BENCHMARK

No	**Formulation**	**D**	**RS**	**F_{min}**
F1	$f(x)= \sum_{i=1..n} \lvert x_i \rvert$	5 15 30	[-100,100]	0
F2	$f(x)=(9x_2^2+2\,x_2^2-11)^2 +(3x_1^2+4\,x_2^2-7)^2$	2	[-100,100]	0
F3	$f(x)=(x_1+10x_2)^2+5(x_3+x_4)^2+(x_2+2x_3)^4+10\,(x_1+x_4)^4$	4	[-100,100]	0
F4	$f(x)=2x_1^2+3x_2^2+4(x_1\,x_2)-6x_1-3x_2$	2	[-100,100]	0

F5	f(x)=$-3803.4-138.08\ x_1-232.92\ x_2+123.08\ x_1^2+203.64\ x_2+182025\ x_1\ x_2$	2	[-100,100]	-3833.12
F6	f(x)=xTx	2	[-100,100]	0

The OMF algorithm was applied to the above test problems and the results are shown in Table VI. The results of OMF are compared to those of GHS[20]. We can clearly see that the two approaches performed comparably. They can find the global optimum solution for all the benchmark problems except for F4. However, when OMF is more precise (for a smaller ε), the algorithm can find the global optimum.

TABLE VI. RESULTS FORINTEGERPROGRAMMING PROBLEMS

Fct	**GHS**	**OMF**

F1(5)
F1(15)
F1(30)
	0(0)
	0(0)
	0(0)
		0(0)
		0(0)
		0(0)

F2	0(0)	0(0)
F3	0(0)	0(0)
F4	-5(1)	-6(0)
F5	- 3833.12(0)	-3833,12(0)
F6	0(0)	0(0)

5. Conclusion:

This work proposed a novel optimization algorithm inspired by image processing tools. Indeed, our algorithm of Optimization by Morphological Filters (OMF) is inspired by the morphological transformations more precisely of the erosion for the search of the global optimum in a multidimensional space. Thirteen test functions were employed in order to benchmark the performance of the proposed algorithm in continuous case. The results showed that OMF was able to provide highly competitive results compared to well-known heuristics such as PSO, GSA,GHS and GWO. For the integer programming problem, OMF was benchmarked with six functions and it showed that it was able to reach the global optimum of the used functions.

We do not pretend having a universal solution for optimization problems because such a solution doesn’t exist but we can say that we developed a new optimization algorithm based on a new approach. We actually are testing the OMF algorithm for the resolution of constrained problems and engineering problems, the results should be communicated in the near future.

References

[1] J.H. Holland(1973).Genetic Algorithms and the optimal allocation of trials, *SIAM Journal of Computing.* Vol. 2, N° 2, pp. 88-105.

[2] A. Colorni, M. Dorigo, V. Maniezzo. Distributed Optimization by Ant Colonies,*Proceedings of the 1rst European Conference on Artificial Life*,pp 134-142.

[3] D. Karaboga(2005), An idea based on honeybee swarm for numerical optimization,Technical Report TR06, *Erciyes University, Engineering Faculty*, Computer Engineering Department.

[4] J. Kennedy and R. Eberhart(1995), Particle swarm optimization, in *Neural Networks 1995 Proceedings, IEEE International Conference*, pp. 1942-1948.

[5] Yang and SuashDeb(2009), Cuckoo Search via Levy flight, *Proceeding Of World Congress on Nature & Biologically Inspired Computing . IEEE Publications*, pp. 210 – 214.

[6] X.-S. Yang(2010), A new metaheuristic bat-inspired algorithm, in *Nature inspired cooperative strategies for optimization*, ed: Springer, pp. 65-74.

[7] S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey Wolf Optimizer, *Advances in Engineering Software* , Vol. 69, pp. 46-61, 2014.

[8] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi(1983), Optimization by simulated annealing, *Journal of Science*, Vol. 220, N° 4598, pp. 671-680,.

[9] Z. W. Geem, J. H. Kim and G. V. Loganathan(2001). A New Heuristic Optimization Algorithm: Harmony Search. Simulation. Vol. 76, N° 2, pp. 60-68,.

[10] E. Alba and B. Dorronsoro(2005),The exploration/exploitation tradeoff in dynamic cellular genetic algorithms,*Evolutionary Computation, IEEE Transactions on,* vol. 9, pp. 126-142.

[11] R. Storn and K. Price(1997), Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, *Journal of global optimization*, vol. 11, pp. 341-359.

[12] X. Yao, Y. Liu and G. Lin(1999), Evolutionary programming made faster,*Evolutionary Computation, IEEE Transactions*, Vol. 3, pp. 82-102.

[13] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi(2009), GSA: a gravitational search algorithm, *Information sciences*, Vol. 179, pp. 2232-2248.

[14] O. K. Erol and I. Eksin(2006), A new optimization method: big bang–big crunch,*Advances in Engineering Software*, vol. 37, pp. 106-111.

[15] A. Kaveh and S. Talatahari(2010), A novel heuristic optimization method: charged system search,*Acta Mechanica*, vol. 213, pp. 267-289.

[16] R. A. Formato(2007), Central force optimization: A new metaheuristic with applications in applied electromagnetics,*Progress In Electromagnetics Research*, Vol. 77, pp. 425-491.

[17] B. Alatas(2011), ACROA: Artificial Chemical Reaction Optimization Algorithm for global optimization,*Expert Systems with Applications*, vol. 38, pp. 13170-13180.

[18] A. Hatamlou(2012), Black hole: A new heuristic optimization approach for data clustering, *Information sciences*.

[19] F. Farrahi-Moghaddam(2007), Curved Space Optimization (CSO): A novel approach to heuristic optimization, *Proceeding of 12th CSICC2007*, Iran.

[20] M.G.H. Omran and M. Mahdavi(2008), Global-best harmony search, *Appl. Math. Comput.*Vol. 198, Issue 2, pp 643-656.

[21] M. Mahdavi, M. Fesanghary, E. Damangir(2007), An improved harmony search algorithm for solving optimization problems, Applied Mathematics and Computation,pp. 1567–1579.

[22] G. Matheron (1967), Elements Pour une Theorie des Milieux Poreux. Paris, France: Masson.

[23] J. Serra(1982), Image Analysis and Mathematical Morphology, *Academic Press*, New-York.

[24] J.G. Postaire and C.P.A. Vasseur (1981), An Approximate Solution to Normal Mixture Identification with Application to Unsupervised Pattern Classification, IEEE Trans. Patt. Anal. & Machine intel. Vol. PAMI-3, n°2, pp. 163-179.

[25] Q. He and L. Wang(2007), An effective co-evolutionary particle swarm optimization for constrained engineering design problems,*Engineering Applications of Artificial Intelligence*, vol. 20, pp. 89-99.

A Combined Data Mining and Multi-Objective Tabu Search approach for Single Customer Dial-a-Ride Problem

A. C. Morais[1], L. Torres[1], T. Galvão Dias[1,2], P. J. S. Cardoso[3] and H. C. L. Fernandes[4]

[1] FEUP - Faculty of Engineering of University of Porto
up201302845@fe.up.pt, up201304426@fe.up.pt, tgalvao@fe.up.pt
[2] INESC TEC - INESC Technology and Science
[3] LARSyS, University of the Algarve
pcardoso@ualg.pt
[4] Yellowfish Travel, Lda
hortensio@yellowfishtransfers.com

Keywords: Dial-a-Ride Problem, Tabu Search Tourism Forecasting, Airport Transfers

Abstract. The Dial-a-Ride problem has been one of most studied routing problems since its appearance in the 70's. The current paper compares a Tabu Search and a parallel insertion algorithm for a Single Customer Dial-a-Ride problem with heterogeneous fleet and drivers, using data mining techniques that enable the prediction of input parameters. Being a single customer problem, the satisfaction of one request at a time is an implied condition. The problem is also characterized by the different capacities of the fleet and the drivers' commission, being the last one based on the number and distance of served requests. These conditions lead to a multi-objective function minimizing the total traveled distance and empty seats in the trips and maximizing the drivers' salaries homogeneity. In addition, the seasonality implied by the fact that the considered data regards a beach tourism destination and the reservations antecedences, which is subject to a high variance, lead to the need of predicting the demand. The proposed approaches are applied to a real world airport private transfer company.

1 Introduction

The Dial-a-Ride problem has been studied for several years mainly concerning the transportation of elderly and disabled people [2]. This research approaches a different type of problem that emerges in the touristic sector, more specifically in airport transfer companies that transport tourists with previous booking to and from the airport. As the case study deals with a beach destination, it is subject to a high seasonality, with a large variance in the number of departure and arrival services throughout one day, as sketched in Figure 1. Additionally, the number of daily requests is also subject to an increasing variability as shown in Figure 2.

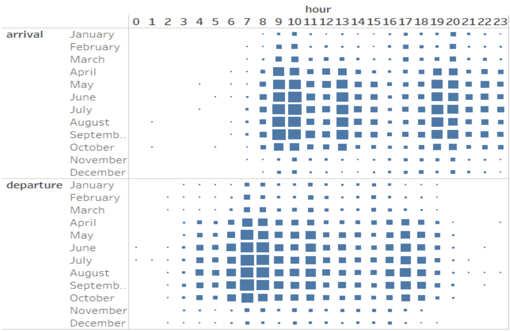


Fig. 1. Map depicting the number of records by type of request and month vs. hour of day.

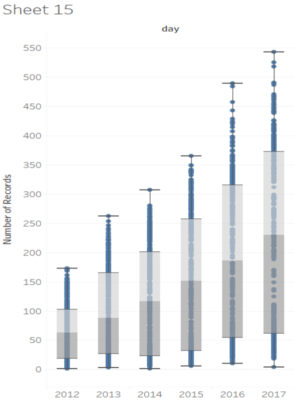


Fig. 2. Boxplots depicting the number of daily requests by year.

2 Problem Formulation

The problem is formulated in a directed gragh $G = (V, A)$. Each request $i \in R = \{1, 2, ..., n\}$ has a pickup $p_i^+ \in P^+ = \{1, 2, ..., n\}$, a drop-off $p_i^- \in P^- = \{n+1, ..., 2n\}$, a load $\sum q_{i,l}$ where l is the type of passenger (baby, child or adult) and either an arrival time e_i or a departure time d_i. The time specified in the request is related with the type of request (arrival or departure) which is given by the binary parameter T_i. Moreover, $V = P^+ \bigcup P^- \bigcup D$, where $D = \{2n+1, ..., 2n+m\}$ is the set of depot points. In this model there are as many depots as drivers since a vehicle starts and ends at a driver's home. Each vehicle $k \in K$ of capacity C_k is allocated at least to one driver, this information is given by the binary parameter $\omega_{j,k}$, where $j \in M = \{1, 2, ...m\}$. Drivers' working time window $[st_j, ed_j]$ and maximum workload Max_j are also parameters of the model.

The model uses two binary three-index variables $x_{g,h}^k$ (equal to 1 if arc (g, h), of distance $\delta_{g,h}$, is performed by the vehicle $k \in K$, 0 otherwise) and $y_{i,j}^k$ (equal to 1 if request i immediately precedes request j in vehicle $k \in K$, 0 otherwise). Regarding scheduling issues, ν_g^k indicates at what time vehicle k starts serving vertex $g \in V$. In addiction, the model calculates $\mathit{\Delta}_i^k$, number of seats unused on vehicle k while serving request i ($\mathit{\Delta}_i^k$ equal to 0 when i is not served by k)and s_j the cumulative salary of the driver j. In order to calculate s_j the set of services for the vehicle k is needed (A_k). The goal of the problem is expressed by a multi-objective function which comprises the minimization of total distance made by the vehicles (f_1), the maximization of the drivers' wages homogeneity (f_2) and the minimization of the total number of empty seats while satisfying all requests (f_3). Although f_1 and f_3 are linear functions, f_2 is non-linear.

$$\min \ f_1 = \sum_{k \in K} \sum_{g \in V} \sum_{h \in V} \delta_{g,h} x_{g,h}^k \qquad \min \ f_2 = \sum_{j \in M} (s_j - \overline{s_j})^2 \qquad \min \ f_3 = \sum_{i \in R} \sum_{k \in K} \mathit{\Delta}_i^k \tag{1}$$

$$\sum_{k \in K} x_{p_i^+, p_i^-}^k = 1 \quad , \forall i \in R \tag{2}$$

$$x_{p_i^-, p_j^+}^k = y_{i,j}^k \quad , \forall k \in K, \ i, j \in R \tag{3}$$

$$\sum_{h \in P^+} x_{d_k^+, h}^k = 1 \quad , \forall k \in K, d_k^+ \in D \tag{4}$$

$$\sum_{g \in P^-} x_{g, d_k^-}^k = 1 \quad , \forall k \in K, d_k^- \in D \tag{5}$$

$$x_{p_i^+, p_i^-}^k \sum_{l \in J} q_{i,j} + \mathit{\Delta}_i^k = C_k x_{p_i^+, p_i^-}^k, \quad \forall k \in K, \ i \in R \tag{6}$$

$$y_{i,j}^k (\nu_{p_i^+}^k + t_{p_i^+, p_i^-} + t_{p_i^-, p_j^+}) \leq \nu_{p_j^+}^k, \quad \forall k \in K, \ i, j \in R \tag{7}$$

$$\nu_{p_i^+}^k + t_{p_i^+, p_i^-} \leq d_i \times T_i, \forall k \in K, i \in R \tag{8}$$

$$\nu_{p_i^+}^k \geq e_i \times (1 - T_i), \forall k \in K, \ i \in R \tag{9}$$

$$\omega_{j,k} (\nu_{d_k^-}^k - \nu_{d_k^+}^k) \leq Max_j, \forall j \in M, k \in K, \ d_k^-, d_k^+ \in D \tag{10}$$

$$\omega_{j,k} \min(\nu_g^k) \geq st_j, \forall k \in K, g \in V, \ j \in M \tag{11}$$

$$\omega_{j,k} \max(\nu_g^k) \leq ed_j, \forall k \in K, g \in V, \ j \in M \tag{12}$$

$$\omega_{m,k} \times s_m = \sum_{i \in A_k} rev_i, \forall k \in K, \ m \in M \tag{13}$$

Constraints (2) and (3) ensure that all requests are satisfied by one vehicle and that the vehicle which picks-up the costumer is the same that drops him off. Equations (4) and (5) ensure that a vehicle starts and ends its daily service at a depot. Regarding vehicles' capacity restrictions, those are verified through constraint (6). Another important group of constraints in a DARP problem concerns the time specifications. Constraint (7) ensures that none of the requests of a vehicle overlap in time. Equations (8) and (9) address costumer's impositions. More precisely a customer either define at what time wants to be picked-up from a place (p_i^+) or dropped-off (p_i^-) and the trip time ($t_{p_i^+, p_i^-}$) is calculated. Therefore, (8) satisfies requirements of the first type and (9) those of the second. The last group of constraints cover drivers' issues, namely, equations (10), (11) and (12) ensure that a driver only works within a previously defined time window. Equation (13) is related with the salaries of drivers being, rev_i the drivers' revenue associated to each request.

3 Algorithm Approach

Typically, an airport transfer company solves the allocation problem manually, in a daily basis. In order to find a faster and more efficient solution for the proposed model, a multi-objective

Tabu Search meta-heuristic is proposed. Since it was proposed by Glover in 1986, Tabu Search has been applied multiple times in routing and Dial-a-Ride problems[1] [3]. Parallel insertion, a simpler heuristic, is also used to solve the problem [3]. Being the beach tourism sector subject to high seasonality, the instances' size range is large. In this particular case study, the size varies between around 30 and 350 requests. This may have an impact in the algorithm running time, so the comparison between the two heuristics proposed may show interesting results or even lead to an hybrid method using both techniques depending on the instance size.

To run the algorithm it is necessary to know in advance both drivers' and vehicles' availability and some additional attributes that characterize the services to be performed, as the day and hour, the pickup and drop off places and the number of passengers. To enable the optimal allocation of the vehicles and drivers to the services to be performed, it is crucial to previously ensure an adequate quantity of available resources. Therefore, predicting the demand to which the company will be subject in the future is of utmost importance.

4 Data Mining Techniques

Forecasting tourists arrivals has been studied for the past decades with increasing importance for destination management companies. A review of works published between 1980-2011 partitions the quantitative tourism forecasting methods in basic and advanced time-series, static and dynamic econometric models and artificial intelligence methods [4]. More recently, artificial intelligence methods have been gaining reputation. When predicting tourists arrivals, the use of data mining techniques, as k-nearest-neighbour, instead of statistical approaches like linear regression, have proved to be advantageous to the prediction's accuracy and to the identification of non-linear relationships [5]. However, the strength of a model is dependent on factors such as the destination being forecasted, data frequency, number of explanatory variables and forecasting horizon [4].

In this specific case, after data cleaning and an overall exploratory analysis, models to forecast the number of requests according to the data frequency are applied. The company at study provided a rich dataset with detailed information concerning six years of operation, which resulted in thousands of records that have been subject to an yearly increase of about 20%.

5 Conclusions

This research addresses the use of computational methods to solve a multi-objective allocation problem with the objectives that include the optimization of the operation procedure and of the homogeneity of collaborators' incomes.

Moreover, the results of the demand prediction combined with the proposed algorithm allow the airport transfer company to continuously compare the orders that have already been booked with the prediction, ensuring that the resources that they possess are adequate. This comparison increases the awareness on how demand is being fulfilled and the capacity of efficiently planning future needs.

References

1. Aldaihani, M. and M. M. Dessouky (2003). "Hybrid scheduling methods for paratransit operations." Computers & Industrial Engineering 45(1): 75-96.
2. Cordeau, J.-F. and G. Laporte (2007). "The dial-a-ride problem: models and algorithms." Annals of Operations Research 153(1): 29-46.
3. Ho, S. C., W. Y. Szeto, Y.-H. Kuo, J. M. Y. Leung, M. Petering and T. W. H. Tou (2018). "A survey of dial-a-ride problems: Literature review and recent developments." Transportation Research Part B: Methodological.
4. Peng, Bo, Haiyan Song, and Geoffrey I. Crouch. (2014). A Meta-Analysis of International Tourism Demand Forecasting and Implications for Practice. Tourism Management 45: 181193.
5. Wolfram, H., Ernesti, D., Fuchs, M., Kronenberg, K., and Lexhagen, M. (2017). "Big Data as Input for Predicting Tourist Arrivals." Information and Communication Technologies in Tourism 2017, pages 187-199.

Fitting epidemiological models' parameters via multi-objective optimization

M. R. Ferrández[1], B. Ivorra[2], J. L. Redondo[1],
A. M. Ramos[2], and P. M. Ortigosa[1]

[1] Dept. of Informatics, Universidad de Almería, ceiA3, Ctra. Sacramento, La Cañada de San Urbano, 04120 Almería, Spain
mrferrandez@ual.es, jlredondo@ual.es, ortigosa@ual.es

[2] Instituto de Matemática Interdisciplinar (IMI) & Dept. de Matemática Aplicada, Universidad Complutense de Madrid, Plaza de las Ciencias, 3, 28040 Madrid, Spain
ivorra@mat.ucm.es, angel@mat.ucm.es

1 Introduction

The design and calibration of epidemiological models that fit the observed behaviour of the diseases are of major interest due to its doubtless applications in real-life situations [4]. One of the most valuable information that these models can provide is about the future evolution of the outbreaks allowing to predict the spread and the impact of diseases. Based on these predictions, the authorities can organise the control measures and the medical resources to prevent some critical situations. Consequently, guaranteeing the accuracy of the model is crucial. However, many epidemiological parameters are usually time and country dependent, such that finding their suitable values is a non-trivial task.

In this work, we propose a multi-objective methodology for determining the parameters of an epidemiological model using some real data about the evolution of the disease in the country. More precisely, we consider two objective functions to minimize: (i) the relative error committed in the detected cases, and (ii) the relative error in the death cases. Those functions measure the difference between the number of cases (resp., deaths) predicted by the epidemiological model and the number of cases (resp., deaths) observed in the reality.

2 Epidemiological model

The epidemiological model considered in this work to illustrate our methodology is the so-called Between-Countries Disease Spread (Be-CoDiS) [3]. It is a compartmental model based on SEIRHD model [2], in which six disjoint states are defined for classifying the individuals in a country:

- Susceptible: The person is not infected by the disease pathogen.
- Infected: The person is infected by the disease pathogen but he/she cannot infect other people since he/she has no visible clinical signs.
- Infectious: The person can infect other people and he/she starts developing clinical signs.
- Hospitalized: The person is hospitalized but he/she can still infect other people.
- Recovered: The person has survived the disease, is no longer infectious and develop a natural immunity to the disease pathogen.
- Dead: The person has not survived the disease. The cadavers of infected people can infect other people until they are buried.

Since we focus on modelling the behaviour of one single country, we have considered a simplified version of the Be-CoDiS model, where the notations $S(t)$, $E(t)$, $I(t)$, $H(t)$, $R(t)$ and $D(t)$ refer to the ratio of people in each state: susceptible, infected, infectious, hospitalized, recovered and dead, respectively; and the population size NP in the considered country is assumed to be constant [3].

In this simplified version, the evolution of the disease in the country has been described by the following system of ordinary differential equations:

$$\begin{aligned}
\frac{\mathrm{d}S(t)}{\mathrm{d}t} =& - S(t)\Big(m_I(t)\beta_I I(t) + m_H(t)\beta_H H(t) + m_D(t)\beta_D D(t)\Big) + \tau E(t) \\
& + \mu_m\Big(E(t) + I(t) + H(t) + R(t)\Big) + \gamma_D(t)D(t), \\
\frac{\mathrm{d}E(t)}{\mathrm{d}t} =& S(t)\Big(m_I(t)\beta_I I(t) + m_H(t)\beta_H H(t) + m_D(t)\beta_D D(t)\Big) \\
& - \left(\mu_m + \gamma_E(t) + \tau\right) E(t), \\
\frac{\mathrm{d}I(t)}{\mathrm{d}t} =& \gamma_E(t)E(t) - \Big(\mu_m + \gamma_I(t)\Big)I(t), \\
\frac{\mathrm{d}H(t)}{\mathrm{d}t} =& \gamma_I(t)I(t) - \Big(\mu_m + \omega\gamma_{HD}(t) + (1-\omega)\gamma_{HR}(t)\Big)H(t), \\
\frac{\mathrm{d}R(t)}{\mathrm{d}t} =& (1-\omega)\gamma_{HR}(t)H(t) - \mu_m R(t), \\
\frac{\mathrm{d}D(t)}{\mathrm{d}t} =& \omega\gamma_{HD}(t)H(t) - \gamma_D(t)D(t),
\end{aligned} \tag{1}$$

where $\mu_m(i)$ is the mortality rate (day^{-1}) of the country meaning the number of deaths per day and per capita, ω is the disease fatality rate (%) defined as the percentage of people in the country who do not survive the disease, τ is the daily rate (%) of the movement of people leaving country, and $\beta_I, \beta_H, \beta_D$ are the disease effective contact rates of people in states I, H, and D, respectively. These disease contact rates are constant parameters depending on the country and representing the mean number of contacts transmitting the disease of a person in the considered state per day before applying the control measures. The functions $\gamma_E(t), \gamma_I(t), \gamma_{HD}(t)$, and $\gamma_{HR}(t)$ also involved in the model are the transition rates from states E to I, I to H, H to D and H to R, respectively. They refer to the number of people moving for one state to another per day and per capita and not only depend on the country but also on time, because the application of the control measures produces variations in the durations of the people in each state. Finally, the functions $m_I(t), m_H(t)$, and $m_D(t)$ describe the efficiency of the control measures.

3 Multi-objective methodology for fitting the epidemiological parameters

Given a set ϕ of values for those epidemiological parameters, as the fatality rate and disease contact rates, the efficiency of the control measures, etc., and given the initial ratios of people in each state $S(0)$, $E(0)$, $I(0)$, $H(0)$, $R(0)$ and $D(0)$ in a country, then, we can obtain the ratios $S^\phi(t)$, $E^\phi(t)$, $I^\phi(t)$, $H^\phi(t)$, $R^\phi(t)$ and $D^\phi(t)$ at each time step by solving numerically the previous System (1). Moreover, we can compute the cumulative number of cases and the cumulative number of deaths, respectively, as:

$$CC^\phi(t) = CC(0) + \int_0^t \gamma_I(t) \cdot I^\phi(t)\mathrm{d}t, \qquad CD^\phi(t) = CD(0) + \omega \int_0^t \gamma_{HD}(t) \cdot H^\phi(t)\mathrm{d}t, \tag{2}$$

where $CC(0)$ and $CD(0)$ are the initial number of cases and of deaths available in the disease reports.

Since our goal is finding a set of values for the parameters of the model that results in a quite accurate description of the disease evolution, we compare to the real data available through the whole evolution of those cumulative numbers in the disease reports: $\{\mathrm{CC}_{\mathrm{real}}(t_j)\}_{j=0}^{j=H}$ and $\{\mathrm{CD}_{\mathrm{real}}(t_j)\}_{j=0}^{j=H}$, where H is the number of available historical data. Thus, the considered multi-objective optimization problem is formulated as follows:

$$\begin{cases} \min\ f_1(\phi), \\ \min\ f_2(\phi), \\ \text{s.t. } \ \phi \in \Phi, \end{cases} \tag{3}$$

where ϕ denotes the set of parameters of the model to be estimated, Φ is the feasible set or search space given by the ranges of those parameters, and $f_1(\phi), f_2(\phi)$ are the objective functions representing the relative errors committed in the cumulative number of detected cases and in the cumulative number of reported deaths, respectively, i.e.:

$$f_1(\phi) = \frac{\left\|\mathrm{CC}_{\mathrm{real}}(t_{\mathrm{f}}) - CC^{\phi}(t_{\mathrm{f}})\right\|_{L^2}}{\left\|\mathrm{CC}_{\mathrm{real}}(t_{\mathrm{f}})\right\|_{L^2}}, \qquad f_2(\phi) = \frac{\left\|\mathrm{CD}_{\mathrm{real}}(t_{\mathrm{f}}) - CD^{\phi}(t_{\mathrm{f}})\right\|_{L^2}}{\left\|\mathrm{CD}_{\mathrm{real}}(t_{\mathrm{f}})\right\|_{L^2}}. \tag{4}$$

Notice that both errors have been computed with the L^2 norm: $\|g(T)\|_{L^2} = \left(\int_0^T (g(t))^2 \,\mathrm{d}t \right)^{1/2}$.

As part of our proposed fitting methodology, we use the multi-objective algorithm called *Weighting Achievement Scalarizing Function Genetic Algorithm* (WASF-GA) [5] for obtaining a set of compromise solutions for Problem 3. Broadly speaking, WASF-GA is an algorithm based on preferences which uses an achievement scalarizing function and a set of weight vectors to lead the evolution of feasible solutions toward a region of interest meaning a zone of the search space whose points are preferred by the persons who solve the problem. More precisely, we employ the parallel version of WASF-GA described in [1].

4 Preliminary results and conclusions

This fitting methodology has been applied to obtain the parameters of the epidemiological model described in Section 2 for the Ebola virus in those countries where it had more magnitude: Guinea, Liberia and Sierra Leone. In all the cases, the proposed methodology allows finding a set of values for the model that provide a evolution of the disease which is very close to the real one.

Acknowledgements

This research has been funded by grants from the Spanish Ministry of Economy and Competitiveness (TIN2015-66680-C2-1-R and MTM2015-64865P); Junta de Andalucía (P12-TIC301), in part financed by the European Regional Development Fund (ERDF). Juana López Redondo is a fellow of the Spanish “Ramón y Cajal” contract program, co-financed by the European Social Fund. The work has been performed under the Project HPC-EUROPA3 (INFRAIA-2016-1-730897), with the support of the EC Research Innovation Action under the H2020 Programme. This work used EPCC's Cirrus HPC Service (https://www.epcc.ed.ac.uk/cirrus).

References

1. M. R. Ferrández, S. Puertas-Martín, J. L. Redondo, B. Ivorra, A. M. Ramos, and P. M. Ortigosa. High-performance computing for the optimization of high-pressure thermal treatments in food industry. *The Journal of Supercomputing*, pages 1–16, 2018.
2. B. Ivorra, D. Ngom, and A. M. Ramos. Be-codis: A mathematical model to predict the risk of human diseases spread between countries—validation and application to the 2014–2015 ebola virus disease epidemic. *Bulletin of Mathematical Biology*, 77(9):1668–1704, Sep 2015.
3. B. Ivorra, D. Ngom, and A. M. Ramos. Stability and sensitivity analysis of be-codis, an epidemiological model to predict the spread of human diseases between countries. *Biomath Communications Supplement*, 4(1), 2017.
4. S. Ma and Y. Xia. *Mathematical Understanding of Infectious Disease Dynamics*. Lecture Notes Series, Institute for Mathematical Sciences. World Scientific, 2009.
5. A. B. Ruiz, R. Saborido, and M. Luque. A preference-based evolutionary algorithm for multiobjective optimization: the weighting achievement scalarizing function genetic algorithm. *Journal of Global Optimization*, 62(1):101–129, 2015.

Iterated-Greedy-Based Metaheuristic with Tabu Search and Simulated Annealing for Solving Permutation Flow Shop Problem

Khadija Mesmar, Said Aqil, and Karam Allali

Laboratory of Mathematics and Applications, Faculty of Sciences and Technologies, University Hassan II of Casablanca, PO Box 146, Mohammedia, Morocco
mesmarkhadija@gmail.com s_aqil@hotmail.fr allali@hotmail.com

Abstract. This work deals with the iterated greedy based metaheuristic with tabu search and simulated annealing for solving permutation flow shop problem. In our work, we will consider a flow shop problem with several jobs and machines and we will calculate the minimum completion time for all the jobs through the all machines (i.e. the makespan). We implement three variant of metaheuristics; the first is iterated greedy with tabu search, the second is iterated greedy with simulated annealing while the last is iterated greedy with both of tabu search and simulated annealing. In addition, we will compare the all with the classical NEH heuristic. Numerical tests show that the iterated greedy based metaheuristic with tabu search gives best results than with simulated annealing.

Keywords. Permutation Flow Shop, Iterated Greedy, Tabu Search, Simulated Annealing, Makespan.

1 Introduction

In the recent decades, the production systems have to face more and more competition in terms of completion time to accomplish the client commands. The decisions of the companies have a significant impact on the manufacture, the cost of the product, the delivery time, the quality, etc. They must optimize the planning and the scheduling of the production which can be considered amongst the most difficult tasks. One of the most thoroughly studied scheduling problems is the flow shop problem (FSP). In the FSP, a set $J = \{J_1, J_2, ..., J_n\}$ of n independent jobs to be processed on a set $M = \{M_1, M_2, ..., M_m\}$ of m machines. Every job $j \in J$, requires a given fixed, non-negative processing time p_{ij} on every machine $i \in M$. Also, all n jobs are to be processed on the m machines in the same order, that is, the jobs follow the same machine order in the shop starting from machine 1 and finishing on machine m. The objective is to find a sequence for processing the jobs in the shop so that a given criterion is optimized. The criterion that is most commonly studied in the literature is the minimization of the total completion time, also called makespan (C_{max}), of the production sequence. A common simplification in the flow shop scheduling problem is to keep the sequence on the first machine maintained throughout the remaining machines. The resulting problem is called the permutation flow shop problem (PFSP); with the makespan criterion it is denoted as $Fm|prmu|C_{max}$ [1]. There are many well-known heuristic approaches for PFSP. For example, Johnsons algorithm[2], CDS's algorithm [3] and Dannenbrings Rapid access (RA) procedure [4] are good examples of constructive methods. NEH heuristic [5] is one amongst the preferable ones for PFSP. To obtain best results for PFSP, some metaheuristic algorithms have been developped. For instance, the iterated greedy (IG) algorithm implemented by Ruiz and Stutzle [6]; tabu search (TS) given by Nowicki and Smutnicki [7] and simulated annealing (SA) by Osman and Potts [8]. In this work, we will implement three variant of metaheuristics; the first is iterated greedy with tabu search, the second is iterated greedy with simulated annealing while the last is iterated greedy with both of tabu search and simulated annealing. Moreover, we will compare the all with the classical NEH heuristic. Numerical tests show that the iterated greedy based metaheuristic with tabu search gives best results than with simulated annealing.

The rest of the work is organized as follows. In Section 2, the permutation ow shop scheduling problem is introduced. In Section 3, we give a detailed description of the iterated greedy with simulated annealing and tabu search algorithm (IGSATS). Computational tests are given and discussed in Section 4. Finally, some conclusions are given in Section 5.

2 Problem statement

In the permutation flow shop scheduling problem, all jobs follow the same order of processing. Our goal is to find a set of compromise solutions so that the makespan is minimized. Flow shop scheduling is a typical assembly line problem where n different jobs have to be handled on different machines. All jobs are processed on all machines in the same order. The processing time of the jobs on the machines is fixed regardless of the order in which the processing is performed. The problem is to sequence n independent jobs $\{J_1, J_2, ..., J_n\}$ on m diffeerent machines $\{M_1, M_2, ..., M_m\}$. Following the same notations proposed by Reeves [9], p_{ij} is the processing time for job j on machine i and a job permutation is represented by the sequence $\pi = \{\pi_1, \pi_2, ...\pi_n\}$. When there are n jobs and m machines, the completion time $C(\pi_i, j)$ is calculated as follows :

$$C(\pi_1, 1) = p(\pi_1, 1), \tag{1}$$

$$C(\pi_i, 1) = C(\pi_{i-1}, 1) + p(\pi_i, 1), \qquad i = 2, ...n \tag{2}$$

$$C(\pi_1, j) = C(\pi_1, j-1) + p(\pi_1, j), \qquad j = 2, ...m \tag{3}$$

$$C(\pi_i, j) = max\{C(\pi_{i-1}, j), C(\pi_i, j-1)\} + p(\pi_i, j), \quad \text{i = 2,...n; j = 2,...m} \tag{4}$$

The makespan is finally defined as:

$$Cmax(\pi) = C(\pi_n, m) \tag{5}$$

Subsequently, the objective is to find a permutation π^* in the set of all permutations Π so that

$$Cmax(\pi*) \leq Cmax(\pi) \quad \forall \pi \in \Pi \tag{6}$$

3 Iterated greedy with simulated annealing and tabu search

In this section, we present different steps of the approximate resolution algorithm IGSATS to solve the flow shop permutation problem (see Algorithm 1). The IGSATS algorithm starts with giving an initial solution (π) generated by the NEH heuristic; after as a second step called a destruction step in which a given number of jobs is randomly selected and removed from the solution without repetition, giving two partial solutions. The first, with the size of tasks, is designated by π_R, including deleted tasks in the order in which they were deleted. The second, with the size n-d jobs, is the original solution without the jobs deleted, which is noted π_D. The third step concerns the construction using the local search criterion, the NEH insertion heuristic is used to complete the solution. The first work $\pi_R(1)$ is inserted into all possible $n - d + 1$ positions in the destroyed solution π_D, thus generating partial solutions $n - d + 1$. Among these partial solutions $n - d + 1$, the best partial solution with the minimum test or the total flow time is chosen and kept for the next iteration. Then, the second job $\pi_R(2)$ is considered and the process continues until π_R is empty or a final solution is obtained. Therefore, π_D is again of size n. The fourth step is the simulated annealing one which is applied as the acceptance criterion because of its excellent performance. The criterion uses a constant temperature that depends on the T_0 parameter of the algorithm:

$$Temperature = T_0 \times \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} p_{ij}}{n \times m \times 10} \tag{7}$$

The fifth step deals with introducing a tabu list that evolves as the mechanism of meta-heuristic tabu search. We consider here a circular tabu list containing sequences of tasks selected in the neighborhood generation process. Before choosing the sequence that will pass the acceptance criterion, we check that it does not appear in the tabu list. The tabu list keeps track of the latest solutions already visited. In the last step, each iteration with the a new tabu list will be updated.

Algorithm 1 Iterated greedy with simulated annealing and tabu search IGSATS

```
$\pi$       $:= \pi_{NEH}$
$\pi_{best}$ $:= \pi_{NEH}$
$TL$    $:= \emptyset$ (TL: Tabu List)
   **While** (termination criterion not satisfied)
     $\pi_D$ := Destruction($\pi$)
     $\pi'$  := Reconstruction($\pi_D, \pi_R$)
     $\pi''$ := Local Search($\pi'$)
     $\beta$  := $exp(-(Cmax(\pi'')-Cmax(\pi))/Temperature)$
        **If** $((TL \neq \emptyset)$  **and**  $(\pi'' \notin LT))$ **then**
           **If** $((Cmax(\pi'') \leq Cmax(\pi))$ **then**
              Insert $\pi$ in TL
              $\pi := \pi''$
                 **If** $((Cmax(\pi) \leq Cmax(\pi_{best}))$ **then**
                    Insert $\pi_{best}$ in TL
                    $\pi_{best} := \pi$
                 **Endif**
            **Elseif** $(random \leq \beta))$ **then**
              Insert $\pi$ in TL
              $\pi := \pi''$
                 **Else**
                    Insert $\pi''$ in TL
           **Endif**
        **Elseif** $((Cmax(\pi'') \leq Cmax(\pi))$ **then**
              Insert $\pi$ in TL
              $\pi := \pi''$
                 **If** $((Cmax(\pi) \leq Cmax(\pi_{best}))$ **then**
                    Insert $\pi_{best}$ in TL
                    $\pi_{best} := \pi$
                 **Endif**
            **Elseif** $(random \leq \beta))$ **then**
              Insert $\pi$ in TL
              $\pi := \pi''$
                 **Else**
                    Insert $\pi''$ in TL
           **Endif**
        **Endif**
   **Endwhile**
  **Return** $\pi_{best}$
```

4 Computational results

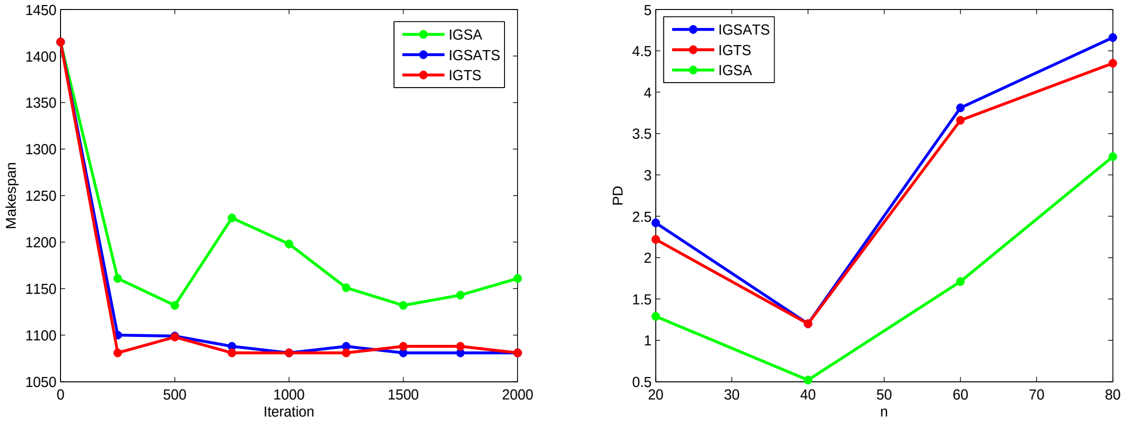
In this section, we provide a comprehensive experimental evaluation and comparison of the proposed IGSATS algorithm with other powerful methods NEH, IGSA[10] and IGTS[11]. For the comparisons we use randomly generated instances where the processing time of the jobs are uniformly distributed between 1 and 99 ranging. The number of jobs is from 20 jobs to 160 and the number of machines is from 5 to 30. In the simulations, we use the same computational conditions including the same computer, the same programming language and the same stopping criteria. All considered algorithms are coded in MATLAB and run on a PC with Intel(R) Core(TM) i5-7200U CPU and 4G RAM. For the computational tests, we set the parameters $d \in$\{3,4,5,6,7,8\} and $T_0 = 0.4$. For all algorithms, the run time is limited to 3600s or until the iterated number equal to 100. The performance of all algorithms is evaluated by a percentage deviation (PD) which is calculated as follows :

$$PD = \frac{C_{max}-Best}{Best} \times 100 \tag{8}$$

To evaluate the performance of our suggested IGSATS algorithm. We will implement three variant of metaheuristics; the first is iterated greedy with tabu search, the second is iterated greedy with simulated annealing while the last is iterated greedy with both of tabu search and simulated annealing. In addition, we will compare the all with the classical NEH heuristic. Computational results are summarized in Table 1 for the number of jobs less than 100 and Table 2 for the number of jobs less more than 100; where C_{max}, CPU and PD represent makespan, percentage deviations and computational time in seconds, respectively.

Table 1. Computational results for random instances.

$n \times m$	NEH	IGSATS			IGTS			IGSA		
	Cmax	Cmax	CPU	PD	Cmax	CPU	PD	Cmax	CPU	PD
20×5	1230	1218	0.11	0.99	1218	0.11	0.99	1218	10.26	0.99
20×10	1567	1530	0.17	2.42	1533	0.17	2.22	1547	11.72	1.29
20×15	1938	1817	0.21	6.66	1831	0.21	5.84	1855	15.17	4.47
40×5	2134	2131	0.31	0.14	2131	0.31	0.14	2133	40.08	0.05
40×10	2689	2657	0.51	1.20	2657	0.52	1.20	2675	156.61	0.52
40×15	3113	3003	0,71	3.66	3014	0.70	3.28	3036	421.87	2.54
60×5	3411	3403	1.26	0.24	3403	1.09	0.24	3403	64.91	0.24
60×10	3681	3546	1.29	3.81	3551	1.27	3.66	3619	101.16	1.71
60×15	4017	3940	2.19	1.95	3941	2.42	1.93	4012	138.51	0.12
80×5	4664	4660	2.55	0.09	4660	2.62	0.09	4660	2.76	0.09
80×10	5626	5512	8.49	2.07	5532	4.46	1.70	5609	287.44	0.30
80×15	4992	4855	6.52	2.82	4855	6.67	2.82	4986	46.11	0.12

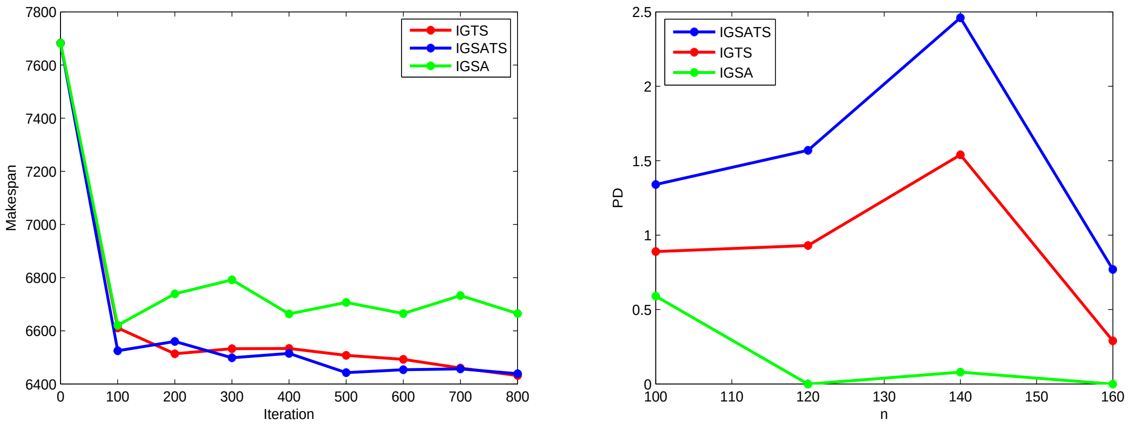


,

Fig. 1. Plots of the Makespan vs number of iterations for 5 machines and 20 jobs (left) and percentage deviation vs the number of jobs for 10 machines (right).

Table 2. Computational results for random instances.

$n{\times}m$	NEH	IGSATS			IGTS			IGSA		
	Cmax	Cmax	CPU	PD	Cmax	CPU	PD	Cmax	CPU	PD
100×10	5657	5603	4.71	0.96	5610	4.56	0.84	5649	362.41	0.14
100×20	6528	6503	9.22	0.38	6504	8.63	0.37	6512	253.19	0.25
100×30	7492	7393	15.95	1.34	7426	16.71	0.89	7448	302.07	0.59
120×10	6852	6764	9.99	1.3	6765	10.23	1.29	6818	198.23	0.50
120×20	7757	7629	19.29	1.68	7666	10.68	1.19	7740	200.93	0.22
120×30	8610	8477	28.98	1.57	8531	28.88	0.93	8610	3600	0.00
140×10	7673	7599	13.45	0.97	7644	13.48	0.38	7661	166.29	0.16
140×20	8579	8440	25.79	1.65	8499	26.01	0.94	8553	357.21	0.30
140×30	9773	9538	39.40	2.46	9625	39.81	1.54	9765	144.14	0.08
160×10	8816	8769	18.07	0.54	8769	18.79	0.54	8802	167.02	0.08
160×20	9966	9831	35.68	1.37	9876	34.39	0.91	9965	483.77	0.01
160×30	10703	10621	51.05	0.77	10672	52.56	0.29	10703	3600	0.00



,

Fig. 2. Plots of the Makespan vs number of iterations for 20 machines and 100 jobs (left) and percentage deviation vs the number of jobs for 30 machines (right).

To show convergence performance, convergence graphs in Figures 1,2. It can be obviously seen that IGSATS algorithm can converge faster than IGTS and IGSA. To evaluate the robustness of three algorithms, changes of percentage deviations are depicted in Figure 1 and 2. It can be seen that IGSA and IGTS algorithms has smaller percentage deviations than IGSATS. Numerical tests show that the iterated greedy based metaheuristic with tabu search gives best results than with simulated annealing.

5 Conclusion

We have studied in this work, the iterated greedy based metaheuristic with tabu search and simulated annealing for solving permutation flow shop problem. We have considered a typical flow shop problem with n jobs and m machines. Our goal is to compute the makespan. We have implemented three variant of metaheuristics; the first is iterated greedy with tabu search, the second is iterated greedy with simulated annealing while the last is iterated greedy with both of tabu search and simulated annealing. In addition, we will compare the all with the classical NEH heuristic. Numerical tests shown that the iterated greedy based metaheuristic with tabu search gives best results than with simulated annealing.

References

1. Pinedo, M., 2002. Scheduling: Theory, Algorithms and Systems, second ed. Prentice Hall, New Jersey.
2. Johnson, S. M. (1954). Optimal two- and three-stage production schedules with setup times included. Naval Research Logistics Quarterly,1(1), 61–68.
3. Campbell, H. G., Dudek, R. A., & Smith, M. L. (1970). A heuristic algorithm for the n job, m machine sequencing problem. ManagementScience, 16(10), 630–637.
4. Dannenbring, D. G. (1977). An evaluation of flow shop sequencing heuristics. Management Science, 23(11), 117–1182.
5. Nawaz M, Enscore EE, Ham I. A heuristic algorithm for the m-machine, n-job ow-shop sequencing problem. Omega. 1983;11:91–95.
6. Ruiz R, Sttzle T. A simple and eective iterated greedy algorithm for the permutation owshop scheduling problem. Eur J Oper Res. 2007;177:2033–2049.
7. Nowicki E, Smutnicki C. A fast tabu search algorithm for the permutation ow-shop problem. Eur J Oper Res. 1996;91:160–175.
8. Osman I, Potts C. Simulated annealing for permutation ow-shop scheduling. Omega. 1989;17:551–557.
9. Reeves CR. A genetic algorithm for owshop sequencing. Comput Oper Res. 1995;22:5–13.
10. Fernandez-Viagas, V., Valente, J. M., & Framinan, J. M. (2018). Iterated-greedy-based algorithms with beam search initialization for the permutation flowshop to minimise total tardiness. Expert Systems with Applications, 94, 58-69.
11. Ding, J. Y., Song, S., Gupta, J. N., Zhang, R., Chiong, R., & Wu, C. (2015). An improved iterated greedy algorithm with a Tabu-based reconstruction strategy for the no-wait flowshop scheduling problem. Applied Soft Computing, 30, 604-613.

On VNS-GRASP and Iterated Greedy Metaheuristics for Solving Hybrid Flow Shop Scheduling Problem with Uniform Parallel Machines and Sequence Independent Setup Time

Said Aqil and Karam Allali

Laboratory Mathematics and Applications, University Hassan II of Casablanca
FST, PO Box 146, Mohammedia, Morocco
s_aqil@hotmail.fr, allali@hotmail.com

Abstract. In this paper, we present some effective metaheuristics for solving hybrid flow shop scheduling problem with uniform parallel machines and sequence independent setup time. In this case, we implement three metaheuristics, the variable neighborhood search algorithm, the greedy random adaptive search procedure and the iterative greedy algorithm. Our problem requires a preparation time of each machine for the processing of a new job. In fact, this time represents the setting or maintenance time specific to the machine independent of the job being processed, this constraint is often encountered in modern production workshops. Our problem is inspired by a real case of paint production workshop in three stages, whose objective function is the minimization of the total flow time taking into account the washing and cleaning constraints as well as the adjustment of the parameters of the machines course of use. For each metaheuristic we choose the appropriate parameters to obtain the optimal solution by exploring the space of the neighborhood of the current solution for the different permutations. We conducted a simulation study on a set of randomly generated instances in order to test the effectivenes of different metaheuristics. We found that the iterative greedy algorithm gives good results compared to the other two metaheuristics.

Keywords: hybrid flow shop, uniform machines, heuristics, meta heuristics, variable neighborhood search, iterative greedy algorithm, greedy random adaptive search procedure, total flow time.

1 Introduction

The tough competition in the industrial market between the company is pushing decision makers to improve their production processes by implementing the new tools of modern technology. The workshops are equipped with machines whose performance can fall over time, to meet the demands of customers on time, the production managers are brought to equip the workshops of new machines with good performance compared to the old machines. The new machine parks therefore require careful organization by optimizing the use of all the functional machines to meet the requirements of the customers. In this context, we propose a simulation study of a flexible flow shop with machine-dependent preparation times also called sequence-independent setup time. Flexible flow shop scheduling problems also known as hybrid flow shop (HF) is widely studied in [1] scheduling literature. The first study concerning this type of problem is presented by [2] by modeling a two-stage flow shop with two identical parallel machines per stage. The problem is qualified as NP hard in the strong sense as soon as the number of machines exceeds two per stage.

The search for the optimal solution by the exact methods is limited for small problems. In the real industrial case the number of jobs is often much higher which pushes the researchers to propose approaches of resolutions based on the heuristics and metaheuristics which give good results in a reasonable time given the complexity of the problem. The first proposed studies generally concern flow shop with identical parallel machines. A notation composed of three fields $\alpha|\beta|\gamma$ is proposed in [3] as presentation of HF problems state art until 1999. We note that a review of the work done in the field of workshop scheduling is presented in [4].

We found that few studies have addressed HF problems with uniform parallel machines, the first works dealing with this type of problem are cited in [5,6]. However, parallel shop flow problems with identical machines [7,8] or unrelated parallel machines [9,10] are largely studied by making the constraints and the criteria to be optimized more diversified. The introduction of sequence independent or sequence-independent setup time in simulation models is of great importance in optimizing flow management. The most recent works [11,12,13] have shown the usefulness of taking this constraint into account in decision-making in the industrial field. This time is considered an unproductive time which generally influences the scheduling of production batches. The mastery and optimal management of this factor allows the company to gain in terms of time and costs generated during the production process. This type of problem is attracting more and more attention from researchers in recent years. They propose models of resolution based essentially on the exact methods for the small problems or meta heuristics for the problems of big size. In the majority of cases, researchers develop models based on meta heuristics for its efficiency and speed to find a good solution to the problem. Many metaheuristics find their application in the resolution of the HF problems we quote in particular the simulated annealing [14,15], the tabu search [16,17], the genetic algorithm [18,19], new algorithms inspired by nature, such as the bee colony algorithm [20,21], the ant colony algorithm [22,23] and the migratory bird algorithm [24,25].

We were inspired by a real case study of production of a paint shop composed of three stages of machines: mixers, grinders and mixers, the machines are uniform parallel after each operation treated on a machine a preparation time sequence independent is needed to process a new job. This type of workshop will be modeled by a hybrid flow shop with uniform machine sequence independent setup time, the objective of which is to minimize the total flow time under the permutation constraint, the problem will be designated by $HF3((QM)_k)_{k=1}^{3}|SIST|\sum_{j=1}^{n} C_{j,3}$. As a result we propose a set of heuristics based essentially on Johnson's rule [26] and the NEH algorithm [27] initially applied to the flow shop problem.

Then the improvement of the solutions is made by three metaheuristics in this case the variable neighborhood loical search algorithm, the iterative greedy algorithm and the greedy random adaptive search procedure. We note that we have chosen these heuristics and its metaheuristics for simplicity and ease in their implementations. The peculiarity of the industrial problem studied in the presence of parallel machines in flow shop with such a constraint allowed us to appreciate the application of the proposed model. The description of the chosen model of the studied workshop is given by the following figure

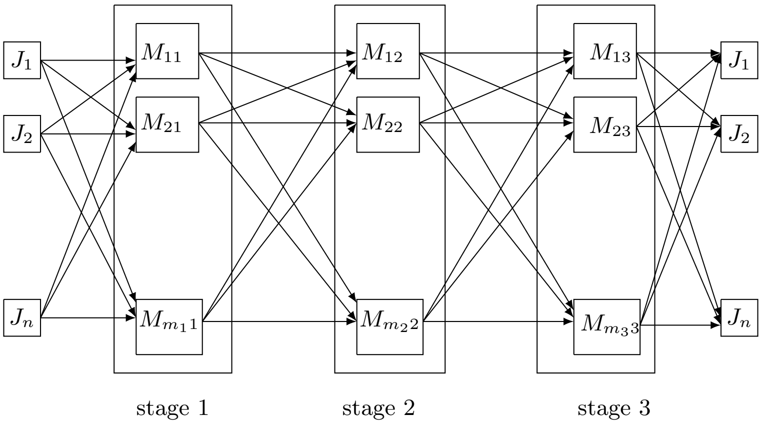


Fig. 1. Three stages hybrid flow shop.

Our paper is organized in the following way, in section 2 we present a description of the proposed model, in section 3 we develop the algorithms of resolution, in section 4 a comparative study is presented between the different algorithms, in section 5 a conclusion completes this work.

2 Description of the hybrid flow shop problem

in the hybrid flow shop workshop scheduling problem a set $J = \{J_1, ..., J_n\}$ of jobs is started in the shop, all jobs follow the same sequence through a set of K stages. On each satge k a machine set $M^{(k)} = \{M_{1k}, ..., M_{m_k k}\}$ is available to handle the same job. A job sequence $\pi = \{\pi_1, ..., \pi_n\}$ is generated in the permutation space of the neighborhood of the current solution is started in the production run of the shop. We note that $v_{ik} \in \mathbb{N}$ represents the speed of the machine i on the stage k and $p_{ijk} = \frac{p_{jk}}{v_{ik}}$, the duration of execution of a job j on a machine i of the stage k. The end date of the job j on a machine i of the stage k is $C_{i,\pi_j,k}$. When the machine is not specified the output date of the jobs of the stage k is noted $C_{\pi_j,k}$. The jobs will be sorted in ascending order of their date $C_{\pi_j,k-1}$ to be processed by the stage k machines obeying the first-come-first-served rule. Either s_{ik} the preparation time independent of the sequence of the machine at stage k before the processing of any job. This time is necessary to prepare the machine for the processing of a new job on the machine i of the stage k. In the proposed model, we note that the sequence-independent setup time is much more important for high-speed machines compared to the lower-speed ones. In fact, the adjustment of uniform machines of high speed usually requires highly skilled managers with low speed skills, which represents the most frequently used industrial cases. In our case the goal is the minimization of the total flow time of all jobs,i.e $\sum_{j=1}^{n} C_{j,3}$ in the last stage. The objective function is determined by a set of equations highlighting the end dates of the jobs, the operating times, the preparation times and the speeds of the machines.

On each stage k a job π_j is assigned to a single machine, when a machine has not processed any job before the completion date is given by the following expression:

$$C_{i,\pi_j,k} = max\{s_{ik}, C_{i,\pi_j,k-1}\} + \frac{p_{jk}}{v_{ik}} \tag{1}$$

If a machine i has already processed a job, the completion date is given by:

$$C_{i,\pi_j,k} = max\{C_{h,\pi_\sigma,k} + s_{ik}, C_{i,\pi_j,k-1}\} + \frac{p_{jk}}{v_{ik}} \tag{2}$$

Where h denotes the machine that has just processed the job π_σ the predecessor of the job on the machine i in the k-stage scheduling. The job is assigned to the machine i^* which completes it as soon as possible and its end date on stage k is given by:

$$C_{i^*,\pi_j,k} = \min_{1 \leq i \leq m_k} \{C_{i,\pi_j,k}\} \tag{3}$$

the objective is the minimization of the sum of the end dates of all the jobs on the last stage named the total flow time(TFT) knowing that:

$$TFT = \sum_{j=1}^{n} C_{j,3} \tag{4}$$

In the problem the objective is to find an optimal permutation π^* which minimizes the total flow time of all the permutations in the search space.

$$TFT(\pi^*) \leqslant TFT(\pi), \pi \in \mathit{\Pi} \tag{5}$$

3 Resolution

Hybrid flow shop problems with setup time is classified NP hard in the strong sense in the scheduling literary. The resolution by heuristics and metaheuristics is often favored by researchers if the size of the problem is too large compared to the exact methods. Here we propose heuristic-based resolution approaches enhanced by highly metaheuristic responses in the optimization domain in this case, the variable neighborhood search algorithm, the iterative greedy algorithm, and the greedy random adaptive search procedure.

3.1 Initialization heuristics

3.1.1 Heuristics Based on Johnson's Rule

To determine an initial solution we apply the Johnson rule used initially for a two-machine flow shop problem that we adopt for a three-stage HF problem. The following algorithm gives the steps for determining the Johnson sequence.

Algorithm 1 Johnson's Rule

```
Input:Two machines : $M^1, M^2$ and the set $J = J_1, ..., J_n$
        The processing time for each job in two machines are $p_j^1, p_j^2$
Step 1: Build the job set $U$ as $p_j^1 < p_j^2$ and sort the jobs in ascending order their processing time on $M^1$.
Step 2: Build the job set $V$ as $p_j^1 \geqslant p_j^2$ and sort jobs by descending order their processing time on $M^2$.
Step 3: Form the Johnson sequence by the following concatenation $\pi_J = UV$.
Output: $\pi_J$
```

We have implemented two heuristics based on the Johnson rule that we adopt for the three-stage HF problem taking into account machine setup time constraints.

- H_1: Calculate the processing times of each job on two virtual machines M^1, M^2 by the following two expressions:

$$p_j^1 = \sum_{k=1}^{2} \min_{1 \leq i \leq m_k} \left[s_{ik} + \frac{p_{jk}}{v_{ik}} \right], \qquad p_j^2 = \sum_{k=2}^{3} \min_{1 \leq i \leq m_k} \left[s_{ik} + \frac{p_{jk}}{v_{ik}} \right] \tag{6}$$

  Construct the sequence by Johnson's rule considering the duration p_j^1, p_j^2.
- H_2: Calculate the processing times of each job on two virtual machines M^1, M^2 by the following two expressions:

$$\tilde{p}_j^1 = \sum_{k=1}^{2} \left[\frac{\sum_{i=1}^{m_k} \left[s_{i,k} + \frac{p_{jk}}{v_{ik}} \right]}{m_k} \right], \qquad \tilde{p}_j^2 = \sum_{k=2}^{3} \left[\frac{\sum_{i=1}^{m_k} \left[s_{ik} + \frac{p_{jk}}{v_{ik}} \right]}{m_k} \right] \tag{7}$$

  Construct the sequence by Johnson's rule considering the durations $\tilde{p}_j^1, \tilde{p}_j^2$.

3.1.2 Heuristics based on the NEH algorithm

We rely on the NEH algorithm to determine a good starting solution. Indeed, this heuristic is considered among the most important heuristics in the scheduling of the flow shop workshops at K machines. We adopt it to solve the parallel machines in flow shop with setup time independent of the sequence. The fundamental stages of this algorithm are described by algorithm 2.

Algorithm 2 NEH algorithm

```
Input: A K machines flow shop and a set$J = J_1, ..., J_n$ and each job $j$ have the processing time $p_{j,1}, ..., p_{jK}$
        in K machines.
Step 1: Calculate the total processing time of each job by: $T(j) = \sum_{k=1}^{K} p_{jk}$
        Construct the sequence $\pi = \{\pi_1, ..., \pi_n\}$ by sorting the jobs in descending order of $T(j)$
Step 2: Schedule the first two jobs $\pi_1$,$\pi_2$ and choose the best sequence $\pi$ of the first two jobs
Step 3: Schedule other jobs
  for $j = 3$ to $n$ do
      Test $\pi_j$ in the different positions of the current sequence built and choose the best sequence with
      the minimum total flow time in all machines and update $\pi_{NEH}$ of the optimal sequence.
  end for
Output: $\pi_{NEH}$
```

- H_3: Determine the smallest total processing time of each job j on the three stages by the following expression:

$$T(j) = \sum_{k=1}^{3} \min_{1 \le i \le m_k} \left[s_{ik} + \frac{p_{jk}}{v_{ik}} \right] \tag{8}$$

  Rank the jobs in descending order to $T(j)$ build an optimal sequence according to the NEH algorithm.
- H_4: Determine the smallest total average processing time of each job j on the three stages by the following expression:

$$\tilde{T}(j) = \sum_{k=1}^{3} \left[\frac{\sum_{i=1}^{m_k} \left[s_{ik} + \frac{p_{jk}}{v_{ik}} \right]}{m_k} \right] \tag{9}$$

  Rank the jobs in descending order $\tilde{T}(j)$ build an optimal sequence according to the NEH algorithm.

Illustration Example: We consider a problem consisting of 5 jobs to be processed on stages, the processing time of jobs are in Table 1. In Table 2 we give the number of machines per stage and the speed and setup time of preparation of each machine on each stage.

Table 1. Job Processing Time in Three Stages.

job	stage 1	stage 2	stage 3
J_1	8	9	10
J_2	9	12	6
J_3	8	6	8
J_4	8	9	4
J_5	8	6	4

Table 2. Setup Time and Machine Speed per Stage.

	stage 1		stage 2		stage 3	
	M_{11}	M_{21}	M_{12}	M_{22}	M_{13}	M_{23}
$Speed : v_{ik}$	1	2	3	1	1	2
$SIST : s_{ik}$	2	3	3	2	3	4

We give here the numerical application for the computation of the parameters of the heuristic H1 and H3.

H_1: To clarify the proposed heuristic, the processing time of each job are calculated on the two virtual machines M^1 and M^2.

$p_1^1 = min(2+\frac{8}{1}, 3+\frac{8}{2}) + min(3+\frac{9}{3}, 2+\frac{9}{1}) = 13$, $p_1^2 = min(3+\frac{9}{3}, 2+\frac{9}{1}) + min(3+\frac{10}{1}, 4+\frac{10}{2}) = 15$

$p_2^1 = min(2+\frac{9}{1}, 3+\frac{9}{2}) + min(3+\frac{12}{3}, 2+\frac{12}{1}) = 14.5$, $p_2^2 = min(3+\frac{12}{3}, 2+\frac{12}{1}) + min(3+\frac{6}{1}, 4+\frac{6}{2}) = 14$

$p_3^1 = min(2+\frac{8}{1}, 3+\frac{8}{2}) + min(3+\frac{6}{3}, 2+\frac{6}{1}) = 12$, $p_3^2 = min(3+\frac{6}{3}, 2+\frac{6}{1}) + min(3+\frac{8}{1}, 4+\frac{8}{2}) = 13$

$p_4^1 = min(2+\frac{8}{1}, 3+\frac{8}{2}) + min(3+\frac{9}{3}, 2+\frac{9}{1}) = 13$, $p_4^2 = min(3+\frac{9}{3}, 2+\frac{9}{1}) + min(3+\frac{4}{1}, 4+\frac{4}{2}) = 12$

$p_5^1 = min(2+\frac{8}{1}, 3+\frac{8}{2}) + min(3+\frac{6}{3}, 2+\frac{6}{1}) = 12$, $p_5^2 = min(3+\frac{6}{3}, 2+\frac{6}{1}) + min(3+\frac{4}{1}, 4+\frac{4}{2}) = 11$

So we have determined the duration of each job: $p_j^1 = [13, 14.5, 12, 13, 12]$, $p_j^2 = [15, 14, 13, 12, 11]$. According to Johnson's while classifying the jobs of U by ascending order on the first machine and the jobs of V in descending order on the second machine on the higher sets is $U = \{3, 1\}$ and $V = \{2, 4, 5\}$ the sequence of Johnson is $\pi_J = \{3, 1, 2, 4, 5\}$.

H_3: In the same way to illustrate this heuristic we calculate the total duration of each job in the three stages, so we have:

$$T(1) = min(2 + \tfrac{8}{1}, 3 + \tfrac{8}{2}) + min(3 + \tfrac{9}{3}, 2 + \tfrac{9}{1}) + min(3 + \tfrac{10}{1}, 4 + \tfrac{10}{2}) = 22$$
$$T(2) = min(2 + \tfrac{9}{1}, 3 + \tfrac{9}{2}) + min(3 + \tfrac{12}{3}, 2 + \tfrac{12}{1}) + min(3 + \tfrac{6}{1}, 4 + \tfrac{6}{2}) = 21.5$$
$$T(3) = min(2 + \tfrac{8}{1}, 3 + \tfrac{8}{2}) + min(3 + \tfrac{6}{3}, 2 + \tfrac{6}{1}) + min(3 + \tfrac{8}{1}, 4 + \tfrac{8}{2}) = 20$$
$$T(4) = min(2 + \tfrac{8}{1}, 3 + \tfrac{8}{2}) + min(3 + \tfrac{9}{3}, 2 + \tfrac{9}{1}) + min(3 + \tfrac{4}{1}, 4 + \tfrac{4}{2}) = 19$$
$$T(5) = min(2 + \tfrac{8}{1}, 3 + \tfrac{8}{2}) + min(3 + \tfrac{6}{3}, 2 + \tfrac{6}{1}) + min(3 + \tfrac{4}{1}, 4 + \tfrac{4}{2}) = 19$$

And so on until you determine all the total duration of all jobs so you get: $T(j) = [22, 21.5, 20, 19, 19]$. The ranking in descending order of $T(j)$ makes it possible to write the sequence to be sequenced by the algorithm of NEH, $\pi_{NEH} = \{1, 2, 3, 4, 5\}$.

In order to visualize the processing of all jobs in the three-stage hybrid flow shop with the constraint of machine sequence independant setup time$HF3((QM)_k)_{k=1}^{3}|SIST|\sum_{j=1}^{n} C_{j,3}$. We consider the Johnson sequence $\pi_J = \{3, 1, 2, 4, 5\}$ that we launch in the production cycle in our workshop. So we visualize in the Gantt chart following the start and end dates of each job on each machine on each stage.

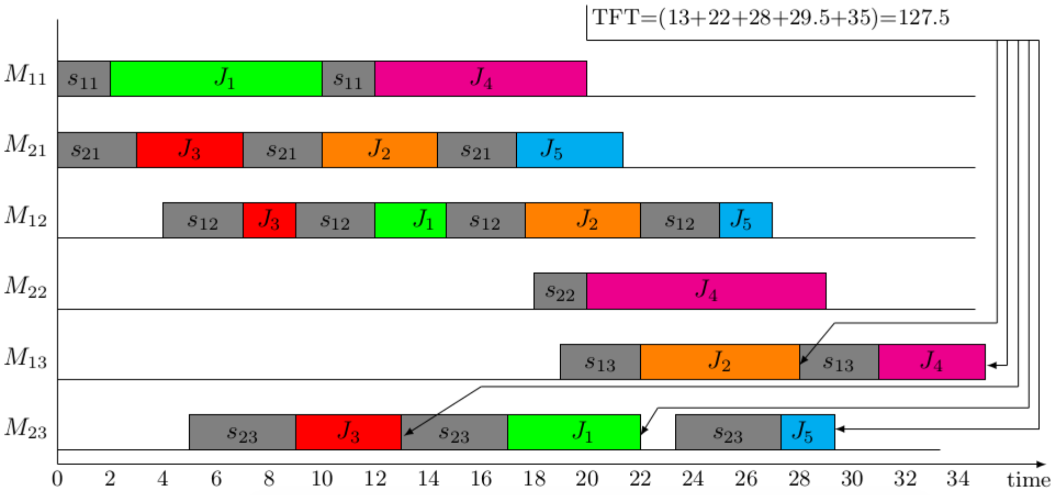


Fig. 2. The Gantt chart for five jobs and two machines per stage.

We can see that $(C_{2,3,3} = 13, C_{2,1,3} = 22, C_{1,2,3} = 28, C_{2,5,3} = 29.5, C_{1,4,3} = 35)$. The total end dates of all job in the last stage represent the total flow time of jobs in the sequence π_J that is $TFT(\pi_J) = 13 + 22 + 28 + 29.5 + 35 = 127.5$ unit time.

3.2 Metaheuristics

To improve the solutions obtained in the initialization phase by heuristics based on the Jonson's rule and the NEH algorithm. We apply three metaheuristics, the variable neighborhood search algorithm, the greedy iterative algorithm, and the greedy random adaptive search procedure. In each metaheuristic, we choose the necessary parameters to adapt it to the problem to be solved. The implementation of this metaheuristic consists in defining a set of neighborhoods based on the methods of permutation and the insertions procedures we describe as follows

N_1 : Randomly select two differents positions of two jobs in the sequence, for example in the sequence $\{2, 4, 5, 6, 1, 3\}$, the positions 2 and 5 jobs swap their positions to obtain the new sequence $\{2, 1, 5, 6, 4, 3\}$.

N_2 : In the insertion neighborhood a position job π_p is drawn at random and a position q is drawn at random. If the position q is to the right of π_p then insert the job at the q position and shift the other jobs to the left. If not, insert the job in the q position and shift the other jobs to the right. For example, in the sequence $\{2, 4, 5, 6, 1, 3\}$ the job π_2 is inserted at position 6, and the other jobs are shifted to the left $\{2, 5, 6, 1, 3, 4\}$.

N_3 : The neighborhood defined by inversion, a block of jobs in an inverse sequence of positions. For example in the sequence $\{2, 4, 5, 6, 1, 3\}$ the block between positions 2 and 5 reverse position giving the new sequence $\{2, 1, 6, 5, 4, 3\}$.

3.2.1 Variable neighborhood search

The variable neighborhood search (VNS) algorithm [28] differs from the local search iterative algorithm by searching in several neighborhoods. Therefore this allows a good exploration in the vicinity of the current solution to reach the right solution. This metaheuristic is one of the most used in the optimization field, applied in this case to the HF scheduling problem [29], we apply it here to solve our problem. The steps of the variable neighborhood search metaheuristic is given by the following algorithm

Algorithm 3 VNS Algorithm

```
Input: $\pi_0 = \{\pi_1, .., \pi_n\}$ The initial sequence
  $\pi^* \leftarrow \pi_0$
  $TFT(\pi^*) \leftarrow TFT(\pi_0)$
  $\pi \leftarrow \pi^*$
  Define the entire neighborhood $N_h (h = 1, ..., h_{max})$
  while {the stopping criterion not satisfied} do
    $h \leftarrow 1$
    while $h \leqslant h_{max}$ do
      Disturbance the current solution and define $\pi'$ in the neighborhood $N_h(\pi')$
      Choose $\pi$” from the neighborhood of $N_h(\pi)$
      if $TFT(\pi”) \leqslant TFT(\pi^*)$ then
        $\pi^* \leftarrow \pi$”
      else
        $h \leftarrow h + 1$
      end if
    end while
  end while
Output: $\pi^*, TFT(\pi^*)$
```

3.2.2 Greedy random adaptive search procedure

The second metaheuristic that we apply in the resolution of this problem is the greedy random adaptive search procedure (GRASP) [30,31]. This metaheuristic proves its effectiveness in solving shop scheduling problems. It is based on two main phases, the construction phase, at the beginning of this phase and starting from an empty initial solution, at each iteration we select candidate jobs from a restricted list of candidates defined in the interval given by the following expression

$$C_{\pi_j,3} \in [C_{mini}, C_{mini} + \alpha \times (C_{maxi} - C_{mini})] \tag{10}$$

Where C_{mini}, C_{maxi} respectively denotes the minimum and maximum end dates in the current sequence build in the third stage. This condition allows to build the short list of candidate jobs. The chosen job from this list will be scheduled in the built-in solution. The second phase consists of looking in the vicinity of the current solution, considering the three neighborhood types cited in the VNS metaheuristic. We reinforce this phase by introducing the model defined by simulated annealing to better explore the studied neighborhood. In the relation defining the restricted condidates of list (RCL), we choose the evaluation function defined by the end date of the jobs on last stage . This function reflects our criterion to be optimized which is the total flow time of in the three stages. The implementation of this metaheuristics requires first of all to choose the type of neighborhood $N(\pi)$ and two settings parameters in this case α and T. The parameter α is the fundamental parameter in the construction phase, it allows to widen or reduce the interval of choice of the list of retained jobs in RCL.

We make the implementation considering several values of α knowing that $\alpha \in \{0.2, 0.4, 0.5, 0.6, 0.8\}$. The T parameter is also a determining factor in the local search phase, the chosen model is given by the following expression

$$T = \lambda \times \frac{\sum_{j=1}^{n} \sum_{k=1}^{K} \min_{1 \le i \le m_k} \left[s_{ik} + \frac{p_{jk}}{v_{ik}} \right]}{10 \times K \times n} \tag{11}$$

This formula is inspired by the model defined in [32] initially planned for the K machine flow shop that we adopt for our three-stages HF problem. We chose to vary the coefficient λ such as: $\lambda \in \{0.9, 0.92, 0.94, 0.96, 0.98\}$
In Algorithm 4 we present the steps of unfolding the metaheuristic (GRASP) with the local search.

Algorithm 4 Greedy Randomized Adaptive Search Procedure with Local Search

```
Input: $\pi = \{\pi_1, .., \pi_n\}$ the initial sequence
  $\pi^* \leftarrow \pi$
  $TFT(\pi^*) \leftarrow TFT(\pi)$
  while {the stopping criterion not satisfied} do
    % construction phase %
    $\Gamma \leftarrow \pi$
    $\pi \leftarrow \emptyset$
    makespan evaluation of the job $\pi_j \in \Gamma$
    while $\Gamma \neq \emptyset$ do
      $C_{mini} \leftarrow min\{C_{\pi_j},\ \pi_j \in \Gamma\}$
      $C_{maxi} \leftarrow max\{C_{\pi_j},\ \pi_j \in \Gamma\}$
      Build the restricted candidates list:
      $RCL = \{\pi_j,\ C_{\pi_j} \in [C_{mini}, C_{mini} + \alpha \times (C_{maxi} - C_{mini})]\}$
      Select a random element $\pi_r$ from the RCL and choose the best position giving the minimum total
      flow time in the constructed solution.
      $\pi \leftarrow \pi \cup \pi_r$
      Update from $\Gamma$
    end while
    % local search phase %
    Choose $\pi'$ from the neighborhood of $N_h(\pi)$
    if $TFT(\pi') < TFT(\pi)$ then
      $\pi \leftarrow \pi'$
      if $TFT(\pi) < TFT(\pi^*)$ then
        $\pi^* \leftarrow \pi$
      end if
    else
      if $random \leqslant exp\{-(TFT(\pi') - TFT(\pi))/T\}$ then
        $\pi \leftarrow \pi'$
      end if
    end if
  end while
Output: $\pi^*, TFT(\pi^*)$
```

3.2.3 Iterative greedy algorithm with local search

The metaheuristic based on the iterative greedy (IG) algorithm mixed with the local search method also consists of two main phases, the phase of obtaining the current solution which is even built of two sub-phases, destruction and construction. The local search phase of exploring the neighborhood of the current solution. We retain the same model presented in the local search phase of the Greedy Random Adaptive Search Procedure. This metaheuristic is frequently used in several scheduling problems [33,34]. We apply it to solve our problem and we adopt its parameters for our case study.

A starting solution is given by one of the heuristics proposed before the objective is to improve it in an iterative way. We varied the starting solution for the different heuristics proposed. This allows us to better check the effectiveness of this metaheuristic. The implementation of this metaheuristic requires two basic settings parameters.

T : It is a parameter that makes it possible to better explore the neighborhood, we adopt the same model retained by the metaheuristic of GRASP.

d : This parameter represents the number of jobs to extract in the destruction phase. We propose a simulation study by varying the number of jobs extracted in the destruction phase by considering $d \in [2, 16]$ depending on the number of jobs started in the production run.

We present in the Algorithm 5 following the detailed steps of determination of the solution obtained by the iterative greedy.

Algorithm 5 Iterated Greedy Algorithm with Local Search

```
Input: $\pi$ sequence obtained by an initialization heuristic.
  $\pi^* \leftarrow \pi$
  $TFT(\pi^*) \leftarrow TFT(\pi)$
  while $\{unsatisfied\ stopping\ criterion\}$ do
    % Destruction phase%
    $\pi' \leftarrow \pi$
    for $h = 1$ to $d$ do
      Extract a job $\pi_h$ at random from the sequence $\pi'$ and add the job $\pi_h$ to the $\Omega$ subset.
    end for
    % Construction phase%
    for $h = 1$ to $d$ do
      Extract the $\pi'_h$ job from $\Omega$ subset.
      Test the job on the different positions in the current sequence $\pi'$ and choose the best position giving
      the smallest total flow time in the last stage.
    end for
    % local search phase%
    Choose $\pi^{''}$ from the neighborhood of $N_k(\pi')$
    if $TFT(\pi^{''}) < TFT(\pi)$ then
      $\pi \leftarrow \pi''$
      if $TFT(\pi) < TFT(\pi^*)$ then
        $\pi^* \leftarrow \pi$
      end if
    else
      if $random \leqslant exp\{-(TFT(\pi^{''}) - TFT(\pi))/T\}$ then
        $\pi \leftarrow \pi^{''}$
      end if
    end if
  end while
Output: $\pi^*, TFT(\pi^*)$
```

4 Numerical simulation

4.1 Simulation instances

In order to validate the proposed metaheuristics, we implement a simulation study based on a set of test problems by varying the number of jobs, the number of machines per stage as well as the parameters of each metaheuristics. These test problems are classified into two broad categories, the first category includes small and medium size problems, the second category concerns larger size problems. This ranking is based on the number of jobs and the number of machines per stage. For the first category, it is characterized by the number of jobs $n \in \{10, ..., 90\}$, the number of machines per stage is $m_k \in \{2, ..., 6\}$; the number of jobs to extract in the metaheuristic of the iterative greedy glutton is $d \in \{2, ..., 6\}$. The processing times p_{jk} are generated according to a uniform law such that $p_{jk} \in [20, 60]$, the setup times are also generated according to a uniform law, knowing that $s_{ik} \in [1, 10]$. For the second category of problems, we consider the number of jobs is defined by $n \in \{100, 120, 140, 160, 180, 200, 220, 240, 260\}$, the number of machines per stage is variable knowing that $m_k \in \{6, ..., 12\}$, the number of jobs to extract is $d \in \{6, ..., 16\}$ for the iterative greedy (IG)algorithm. The processing times are generated according to a uniform law such that $p_{jk} \in [40, 100]$, . We note that the $\alpha \in \{0.4, 0.5, ...0.9\}$ parameter of the greedy random adaptive search procedure (GRASP) will be identical for both categories during the simulation. Speeds of the uniform machines are defined in the set of integers such that $v_{ik} \in \{1, 2, 3, 4, 5\}$ and the setup time is define by $s_{ik} \in [10, 20]$, taking into account the constraints of machine preparation, knowing that when the machine has a high speed it requires a more time of preparation than others.

4.2 Experimental results

Objective of the simulated instances is to propose a comparative study between the different algorithms. The variability of the parameters of each metaheuristic has led us to choose cases highlighting the significant difference between them. The experimental results are very important and promising to verify the effectiveness of the metaheuristics of resolutions our problem. In order to better study their efficiency, we perform two types of analysis.

First analysis consists of comparing the three metaheuristics by calculating the Relative Percentage Deviation (RPD) by the expression following

$$RPD = \left[\frac{TFT - TFT^*}{TFT^*}\right] \times 100 \tag{12}$$

Where TFT designates the value found by a metaheuristic and TFT * the minimum value. For a calibration of the RPD parameter we consider the results for an average of 10 instances for each type of test problem. In this analysis we make the comparison while considering the initial solutions given by the four heuristics (H_1, H_2, H_3, H_4).

Table 3. Experimental results of the performance analysis for all algorithms of the category 1. (the best RPD value is in bold)

Instance			VNS				GRSAP				IG			
$n \times m_1 \times m_2 \times m_3$	TFT*	$Time_1(s)$	H_1	H_2	H_3	H_4	H_1	H_2	H_3	H_4	H_1	H_2	H_3	H_4
$10 \times 4 \times 2 \times 3$	632	40	5.34	4.25	2.33	4.66	4.24	3.55	3.21	3.53	4.88	4.72	3.57	**2.56**
$10 \times 3 \times 3 \times 3$	745	40	2.45	5.44	2.35	3.55	2.78	5.33	2.33	3.22	2.33	5.22	**2.22**	3.21
$10 \times 2 \times 4 \times 4$	625	40	6.13	4.55	**2.33**	4.66	5.24	4.55	5.21	4.53	4.88	2.72	2.57	3.56
$20 \times 2 \times 4 \times 4$	2885	80	**3.23**	4.45	6.25	4.78	3.25	3.25	4.21	3.34	4.55	5.21	3.35	4.21
$20 \times 3 \times 4 \times 4$	2634	80	3.45	6.22	3.45	4.55	3.48	6.11	3.14	4.78	3.25	6.25	**3.11**	4.34
$20 \times 3 \times 5 \times 4$	2478	100	3.45	5.65	6.71	4.41	6.23	4.98	4.65	3.31	**2.78**	4.45	6.15	5.25
$30 \times 2 \times 4 \times 5$	5986	150	4.52	3.31	4.79	5.56	3.27	4.41	5.26	6.12	3.34	**2.52**	4.42	5.21
$30 \times 3 \times 4 \times 5$	5358	150	4.66	5.66	**3.28**	5.25	4.81	5.55	4.23	5.55	4.55	5.51	3.67	5.21
$30 \times 5 \times 3 \times 4$	4358	150	6.11	5.21	3.33	4.25	3.25	2.88	2.68	3.65	4.52	**2.57**	3.34	4.45
$40 \times 5 \times 4 \times 3$	8025	200	5.43	3.56	4.78	**2.89**	3.65	4.45	3.33	2.89	5.23	3.79	4.11	3.25
$40 \times 3 \times 4 \times 5$	8557	200	4.65	4.33	4.81	4.25	**4.22**	4.55	5.21	5.56	4.68	4.41	4.67	4.44
$40 \times 5 \times 4 \times 4$	7854	200	2.23	4.56	6.2	3.58	4.34	5.21	4.45	3.78	3.34	3.35	4.23	**2.11**
$50 \times 3 \times 6 \times 4$	13886	300	4.11	3.44	2.45	6.14	4.78	3.54	**2.21**	3.55	5.22	3.56	2.78	3.45
$50 \times 6 \times 5 \times 5$	12555	300	4.71	5.71	3.33	3.11	4.22	5.55	3.66	**2.87**	4.82	5.71	3.66	3.78
$50 \times 6 \times 6 \times 6$	11567	300	**2.11**	3.36	4.25	3.56	4.11	3.11	4.31	2.12	3.21	4.55	3.45	4.23
$60 \times 6 \times 5 \times 5$	15235	360	4.34	4.67	3.26	2.79	2.66	3.55	4.25	5.34	2.34	**2.23**	3.22	2.68
$60 \times 6 \times 5 \times 6$	14567	360	5.25	5.45	3.66	3.11	5.71	5.33	3.88	4.44	5.34	5.22	4.22	**2.71**
$60 \times 6 \times 5 \times 4$	13563	360	4.56	3.12	2.83	4.45	5.23	3.21	2.65	**2.45**	3.21	4.56	3.67	4.23
$70 \times 5 \times 4 \times 6$	24567	420	5.34	3.45	5.34	2.98	3.45	5.34	2.56	3.24	2.34	**2.22**	3.31	3.23
$70 \times 6 \times 6 \times 5$	22432	420	4.55	4.55	3.62	**2.55**	4.77	4.72	2.99	4.61	4.43	4.43	2.88	2.71
$70 \times 5 \times 6 \times 4$	24589	420	2.23	3.34	2.56	4.42	3.45	3.56	2.45	3.31	3.66	2.12	2.23	**1.63**
$80 \times 4 \times 5 \times 6$	23879	480	2.34	1.84	4.45	3.78	3.67	3.89	2.55	2.67	2.87	3.45	**1.62**	1.87
$80 \times 5 \times 6 \times 5$	23789	480	3.88	5.33	3.44	2.22	3.77	3.41	3.11	2.67	3.71	5.22	2.77	**1.88**
$80 \times 6 \times 6 \times 6$	20456	480	3.45	2.78	3.78	4.23	2.22	3.12	3.45	3.56	2.24	**1.78**	3.67	2.45
$90 \times 6 \times 5 \times 5$	34789	540	1.67	3.55	4.25	2.34	2.56	1.82	3.23	2.45	1.62	1.89	2.23	**1.52**
$90 \times 5 \times 5 \times 6$	33543	540	3.88	4.88	3.12	2.11	3.55	4.22	2.88	2.78	3.44	4.14	2.71	**1.51**
$90 \times 6 \times 6 \times 6$	30564	540	1.56	4.56	3.23	3.76	4.11	3.86	2.57	3.64	2.53	2.29	1.67	**1.54**

Tab.3 and Tab.4 summarize the partial results presented in our paper. In the first column we specify the size of the studied instance, the second column represents the value of the optimal solution, the third column represents the limit time of computes not to be desiccated as a stopping criterion in the three metaheuristics. In the remaining columns, the RPD values for the three metaheuristics $(VNS, GRASP, IG)$ are given according to the four heuristics $(H_1, H_2, H_3, H4)$ used in the initial solutions.

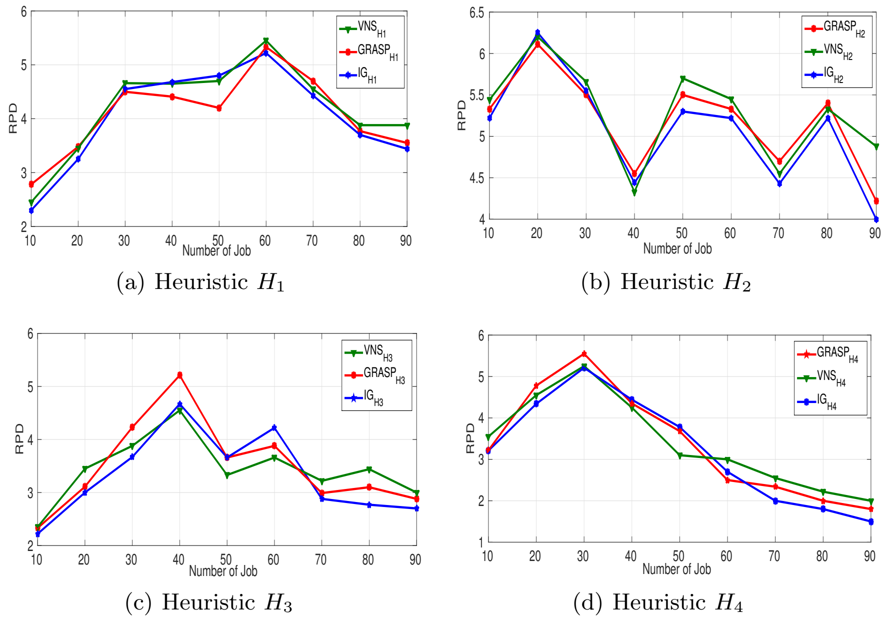
Table 4. Experimental results of the performance analysis for all algorithms of the category 2. (the best RPD value is in bold)

Instance			VNS				GRSAP				IG			
$n \times m_1 \times m_2 \times m_3$	TFT*	$Time_2(s)$	H_1	H_2	H_3	H_4	H_1	H_2	H_3	H_4	H_1	H_2	H_3	H_4
$100 \times 10 \times 7 \times 10$	48645	500	4.34	3.55	2.33	5.66	3.24	3.44	3.11	2.53	3.88	2.12	2.57	**1.56**
$100 \times 11 \times 9 \times 11$	44568	550	2.11	2.22	2.65	2.11	2.12	2.13	2.22	2.31	1.88	1.89	2.11	**1.71**
$100 \times 12 \times 8 \times 12$	40234	600	3.34	2.65	2.63	3.66	2.24	**1.55**	2.21	3.53	2.88	3.72	2.77	2.56
$120 \times 11 \times 9 \times 10$	52347	660	2.56	3.21	**1.62**	2.56	3.64	2.98	1.88	3.23	2.22	1.66	1.88	2.11
$120 \times 9 \times 10 \times 11$	51897	660	2.87	2.99	2.88	2.21	2.44	2.66	2.55	2.81	2.21	2.88	2.76	**1.86**
$120 \times 11 \times 1 \times 11$	50675	660	3.27	2.23	3.34	2.89	3.43	2.78	2.13	3.67	1.88	**1.55**	1.64	2.64
$140 \times 8 \times 11 \times 10$	66345	770	2.78	3.2	32.78	3.56	2.89	2.68	3.21	3.56	3.45	**1.88**	2.22	1.98
$140 \times 10 \times 8 \times 11$	65789	770	3.11	3.44	3.11	3.12	2.88	3.22	2.81	3.11	2.99	3.22	3.61	**2.61**
$140 \times 11 \times 11 \times 11$	60897	770	2.56	3.34	2.65	2.63	2.12	2.34	2.45	2.54	1.95	1.86	**1.68**	2.11
$160 \times 12 \times 8 \times 10$	74678	960	2.89	1.89	2.12	3.11	**1.67**	2.22	1.88	2.45	1.89	2.12	2.22	1.89
$160 \times 12 \times 8 \times 11$	71457	960	**2.66**	2.88	3.83	3.11	2.88	3.11	2.91	2.91	3.11	3.11	3.11	2.81
$160 \times 12 \times 10 \times 12$	68674	960	3.12	2.24	1.77	2.26	2.89	3.15	3.15	3.21	2.26	**1.66**	2.23	1.85
$180 \times 9 \times 12 \times 10$	113456	1080	2.45	2.33	1.67	2.87	2.45	2.55	3.45	2.68	1.85	1.87	**1.66**	2.11
$180 \times 8 \times 12 \times 11$	117567	1080	2.88	2.88	2.88	3.11	**2.53**	2.98	3.35	2.71	2.55	2.55	2.55	2.71
$180 \times 11 \times 7 \times 12$	123456	1080	2.23	2.45	3.11	3.21	2.25	**1.56**	2.43	2.62	3.12	3.25	3.24	2.22
$200 \times 10 \times 12 \times 10$	153876	1200	3.23	3.56	2.86	2.87	2.65	2.75	2.54	2.53	**1.88**	2.31	2.54	2.11
$200 \times 10 \times 12 \times 9$	151324	1200	2.53	2.77	2.55	3.11	2.78	2.88	2.71	2.51	2.44	2.44	**2.11**	2.71
$200 \times 10 \times 8 \times 12$	157657	1200	3.45	2.25	2.45	2.22	**2.11**	3.25	3.45	2.66	2.98	3.15	2.56	2.88
$220 \times 11 \times 11 \times 12$	186987	1320	3.11	3.45	2.65	2.46	2.35	2.74	2.32	2.86	2.21	**2.11**	2.89	2.25
$220 \times 12 \times 10 \times 9$	196346	1320	2.11	2.11	2.34	2.55	2.53	2.51	2.34	2.78	**1.81**	1.88	1.89	2.11
$220 \times 12 \times 12 \times 11$	192456	1320	3.22	2.67	3.21	2.28	**1.83**	1.45	2.45	1.85	2.22	2.16	1.97	1.85
$240 \times 11 \times 10 \times 8$	211786	1440	2.45	2.89	2.15	2.35	2.21	1.87	**1.76**	2.11	2.25	2.76	2.68	2.56
$240 \times 9 \times 12 \times 12$	205452	1440	1.71	1.77	1.88	2.22	2.11	1.77	2.46	2.11	1.66	**1.55**	1.77	1.81
$240 \times 12 \times 12 \times 12$	200342	1440	3.12	2.78	3.45	2.76	1.87	**1.67**	1.86	1.69	1.83	2.22	2.35	2.45
$260 \times 11 \times 8 \times 12$	261234	1560	2.34	2.11	2.89	1.89	2.45	2.22	3.24	2.15	**1.77**	1.86	2.11	2.34
$260 \times 10 \times 12 \times 12$	250064	1560	1.76	1.66	1.66	2.11	1.88	1.55	2.12	1.81	1.66	**1.35**	1.45	1.51
$260 \times 12 \times 12 \times 12$	245164	1560	2.56	1.76	1.86	1.86	1.95	1.85	1.54	**1.44**	1.42	1.88	1.68	1.85

Second analysis consists in studying the speed of convergence towards the optimal solution, for this one will limit the computation time according to the category of the simulated instances. For the first category, the limit calculation time is defined by $Time_1 = n \times max(m_1, m_2, m_3)$. This limits the search for the three metaheuristics in the space of the solutions that are close to the permutation set defined by the neighborhood.In the same way we will limit the calculation time for the second category by $Time_2 = \frac{n}{2} \times max(m_1, m_2, m_3)$.

Fig.3 shows a comparative study between the three metaheuristics for the first category based on the solutions obtained by the four heuristics. We found that for this category the difference between these three metaheuristics is not significant the maximum difference is of 6.25%for all the studied instances. We also find that the metaheuristic IG records the lowest RDP for 22 instances among 36 which represents 61.11%of simulated instances.

In Fig.4, we report the simulation study results of instances of the second category considered large. We found that the greedy iterative algorithm also gives good results compared to other metaheuristics by achieving the lowest value of RPD. Indeed on the 36 instances studied in 26 il achieves the lowest RPD is a rate of 72.22% with the highest RPD values of 1.35%.



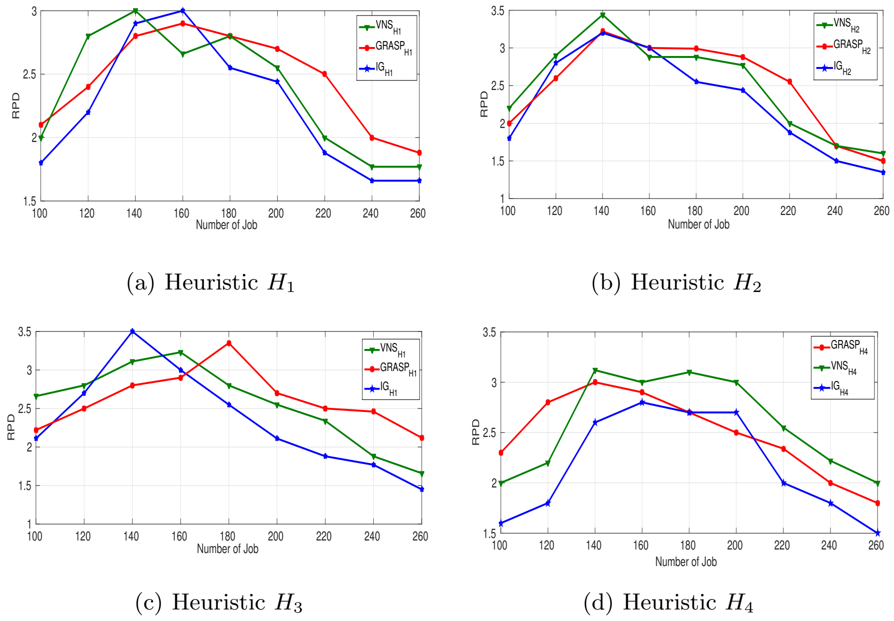
(a) Heuristic H_1

(b) Heuristic H_2

(c) Heuristic H_3

(d) Heuristic H_4

Fig. 3. RPD variation for all algorithms of category 1



(a) Heuristic H_1

(b) Heuristic H_2

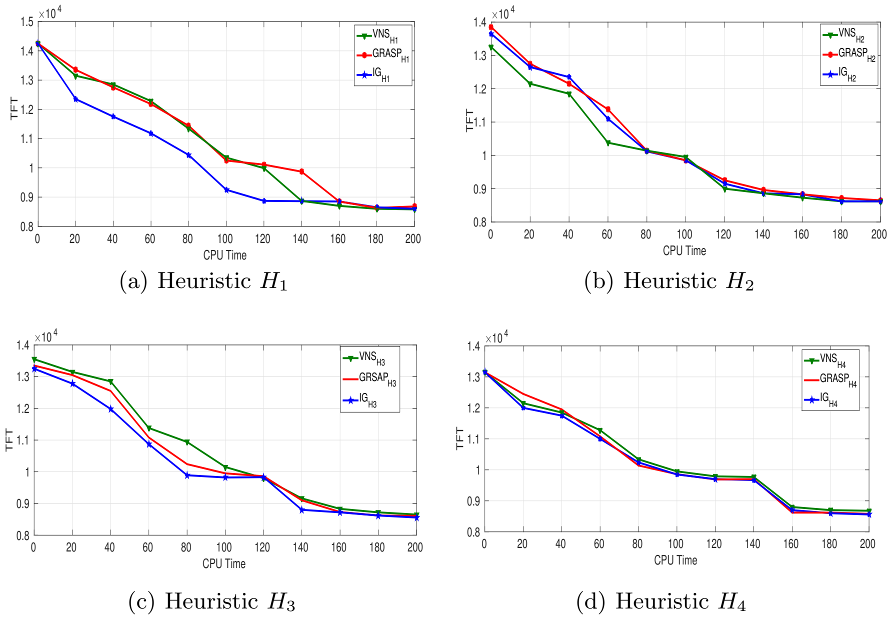
(c) Heuristic H_3

(d) Heuristic H_4

Fig. 4. RPD variation for all algoritms of category 2

The speed of convergence of the heuristics in the search for the optimal solution is one of the most interesting performances to study in this kind of simulation. The second analysis consists in making a comparative study between the different heuristics in terms of time of total flow time of scheduling of all the jobs. This makes it possible to better know the speed of convergence of each metaheuristic to converge towards the optimal solution. We plot the variation of the objective function (TFT) as a function of the CPU time by limiting itself by the stop criterion imposed to no longer explore the whole neighborhood of the current solution.

Our numerical tests are implemented in Matlab 014a on an intel (R) Core (TM) i3 CPU M350 2.27HHz, 2.26GHz computer with a RAM of 6 GB. The interest of the convergence of our metaheuristics in terms of calculation in the search for the optimal solution is a primordial factor in the analysis of the results. The interest of the convergence of our metaheuristics in terms of calculation in the search for the optimal solution is a primordial factor in the analysis of the results. We give the results of two instances representing both categories of problem. We limit ourselves to these two instances which are more than enough to illustrate the difference between the three metaheuristics during the process of finding a solution. Fig.5 shows the evolution of the TFT over the time of the instance $40 \times 3 \times 4 \times 5$ representative of category 1 whose time limit $Time_1$ is 200 seconds. In Fig.6 we show the evolution of the TFT of the instance $160 \times 12 \times 10 \times 12$ whose time limit is $Time_2$ is 960 seconds.



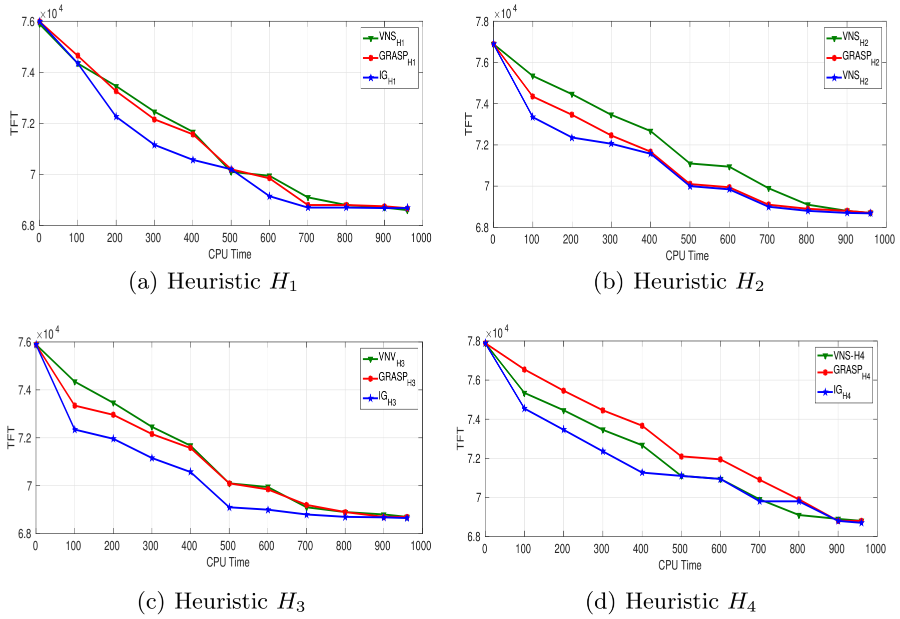
(a) Heuristic H_1

(b) Heuristic H_2

(c) Heuristic H_3

(d) Heuristic H_4

Fig. 5. Evolution of TFT value vs CPU Time for all algoritms of category 1



(a) Heuristic H_1

(b) Heuristic H_2

(c) Heuristic H_3

(d) Heuristic H_4

Fig. 6. Evolution of TFT value vs CPU Time for all algoritms of category 2

From the plots on both Fig.5 and Fig.6 showing the evolution of TFT during the search for the optimal solution. We note that for the four initial heuristics the greedy iterative algorithm gives the best results in terms of the quality of the solution obtained and in terms of the speed of convergence towards this solution. We also find that the best solution obtained based on the H_3 heuristic recording the largest decrease in the initial solution during the search process for the iterative greedy algorithm.

5 Conclusion

In this paper, we have presented three metaheuristics to solve the problem of hybrid workshop flow scheduling with three stages, uniform parallel machine and with sequence-independent setup time. The goal is to minimize the total flow time of all jobs by taking into account the permutation constraints of solutions throughout the neighborhood. The used metaheuristics are the variable neighborhood local search algorithm, the greedy random adaptive search procedure and the iterative greedy algorithm. The complexity of the problem encountered led us to implement four heuristics that will be the basis of three metaheuristics chosen for the resolution of this problem. In fact, the initial solutions obtained by these heuristics are based on Johnson's algorithm and the NEH algorithm that we adopt for our studied problem. Our problem is inspired by a real industrial case representing the production in a painting workshop. We conducted a simulation study on a set of instances modeling industrial encountered cases. Our study was successful in terms of performance recorded by heuristics and metaheuristics applied to solve the problem. The final analysis of the simulation allows us to conclude that the iterative greedy algorithm gives good results compared to the other two metaheuristics the other two metaheuristics based on initial solutions based on the NEH algorithm.

References

1. Pinedo, M. L. (2016). Scheduling: theory, algorithms, and systems. Springer.
2. Gupta, J. N. (1988). Two-stage, hybrid flowshop scheduling problem. Journal of the Operational Research Society, 39(4), 359-364.
3. Vignier, A., Billaut, J. C., & Proust, C. (1999). Les problèmes d'ordonnancement de type fow shop hybride: état de l'art. RAIRO-Operations Research, 33(2), 117-183.
4. Ribas, I., Leisten, R., & Framiñan, J. M. (2010). Review and classification of hybrid flow shop scheduling problems from a production system and a solutions procedure perspective. Computers & Operations Research, 37(8), 1439-1454.
5. Dessouky, M. M. (1998). Scheduling identical jobs with unequal ready times on uniform parallel machines to minimize the maximum lateness. Computers & Industrial Engineering, 34(4), 793-806.
6. Huang, W., & Li, S. (1998). A two-stage hybrid flowshop with uniform machines and setup times. Mathematical and Computer Modelling, 27(2), 27-45.
7. Nowicki, E., & Smutnicki, C. (1998). The flow shop with parallel machines: A tabu search approach. European Journal of Operational Research, 106(2-3), 226-253.
8. Allaoui, H., & Artiba, A. (2004). Integrating simulation and optimization to schedule a hybrid flow shop with maintenance constraints. Computers & Industrial Engineering, 47(4), 431-450.
9. Low, C. (2005). Simulated annealing heuristic for flow shop scheduling problems with unrelated parallel machines. Computers & Operations Research, 32(8), 2013-2025.
10. Yaurima, V., Burtseva, L., & Tchernykh, A. (2009). Hybrid flowshop with unrelated machines, sequence-dependent setup time, availability constraints and limited buffers. Computers & Industrial Engineering, 56(4), 1452-1463.
11. Pan, Q. K., Gao, L., Li, X. Y., & Gao, K. Z. (2017). Effective metaheuristics for scheduling a hybrid flowshop with sequence-dependent setup times. Applied Mathematics and Computation, 303, 89-112.
12. Kia, H., Ghodsypour, S. H., & Davoudpour, H. (2017). New scheduling rules for a dynamic flexible flow line problem with sequence-dependent setup times. Journal of Industrial Engineering International, 13(3), 297-306.
13. Manupati, V. K., Rajyalakshmi, G., Chan, F. T., & Thakkar, J. J. (2017). A hybrid multi-objective evolutionary algorithm approach for handling sequence-and machine-dependent set-up times in unrelated parallel machine scheduling problem. Sadhana, 42(3), 391-403.
14. Nikzad, F., Rezaeian, J., Mahdavi, I., & Rastgar, I. (2015). Scheduling of multi-component products in a two-stage flexible flow shop. Applied Soft Computing, 32, 132-143.

15. Ramezani, P., Rabiee, M., & Jolai, F. (2015). No-wait flexible flowshop with uniform parallel machines and sequence-dependent setup time: a hybrid meta-heuristic approach. Journal of Intelligent Manufacturing, 26(4), 731-744.
16. Shahvari, O., Salmasi, N., Logendran, R., & Abbasi, B. (2012). An efficient tabu search algorithm for flexible flow shop sequence-dependent group scheduling problems. International Journal of Production Research, 50(15), 4237-4254.
17. Wang, S., & Liu, M. (2014). Two-stage hybrid flow shop scheduling with preventive maintenance using multi-objective tabu search method. International Journal of Production Research, 52(5), 1495-1508.
18. Jun, S., & Park, J. (2015). A hybrid genetic algorithm for the hybrid flow shop scheduling problem with nighttime work and simultaneous work constraints: A case study from the transformer industry. Expert Systems with Applications, 42(15-16), 6196-6204.
19. Sukkerd, W., & Wuttipornpun, T. (2016). Hybrid genetic algorithm and tabu search for finite capacity material requirement planning system in flexible flow shop with assembly operations. Computers & Industrial Engineering, 97, 157-169.
20. Pan, Q. K., Wang, L., Li, J. Q., & Duan, J. H. (2014). A novel discrete artificial bee colony algorithm for the hybrid flowshop scheduling problem with makespan minimisation. Omega, 45, 42-56.
21. Kheirandish, O., Tavakkoli-Moghaddam, R., & Karimi-Nasab, M. (2015). An artificial bee colony algorithm for a two-stage hybrid flowshop scheduling problem with multilevel product structures and requirement operations. International Journal of Computer Integrated Manufacturing, 28(5), 437-450.
22. Chen, L., Zheng, H., Zheng, D., & Li, D. (2015, May). An ant colony optimization-based hyper-heuristic with genetic programming approach for a hybrid flow shop scheduling problem. In Evolutionary Computation (CEC), 2015 IEEE Congress on (pp. 814-821). IEEE.
23. Qin, W., Zhang, J., & Song, D. (2018). An improved ant colony algorithm for dynamic hybrid flow shop scheduling with uncertain processing time. Journal of Intelligent Manufacturing, 29(4), 891-904.
24. Niroomand, S., Hadi-Vencheh, A., ?ahin, R., & Vizvári, B. (2015). Modified migrating birds optimization algorithm for closed loop layout with exact distances in flexible manufacturing systems. Expert Systems with Applications, 42(19), 6586-6597.
25. Zhang, B., Pan, Q. K., Gao, L., Zhang, X. L., Sang, H. Y., & Li, J. Q. (2017). An effective modified migrating birds optimization for hybrid flowshop scheduling problem with lot streaming. Applied Soft Computing, 52, 14-27.
26. Johnson, S. M. (1954). Optimal two-and three-stage production schedules with setup times included. Naval Research Logistics (NRL), 1(1), 61-68.
27. Nawaz, M., Enscore Jr, E. E., & Ham, I. (1983). A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. Omega, 11(1), 91-95.
28. Mladenovic, N., & Hansen, P. (1997). Variable neighborhood search. Computers & operations research, 24(11), 1097-1100.
29. Behnamian, J., & Ghomi, S. F. (2011). Hybrid flowshop scheduling with machine and resource-dependent processing times. Applied Mathematical Modelling, 35(3), 1107-1123.
30. Davoudpour, H., & Ashrafi, M. (2009). Solving multi-objective SDST flexible flow shop using GRASP algorithm. The International Journal of Advanced Manufacturing Technology, 44(7-8), 737-747.
31. Alekseeva, E., Mezmaz, M., Tuyttens, D., & Melab, N. (2017). Parallel multi-core hyper-heuristic GRASP to solve permutation flow-shop problem. Concurrency and Computation: Practice and Experience, 29(9).
32. Ruiz, R., & Stützle, T. (2007). A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem. European Journal of Operational Research, 177(3), 2033-2049.
33. Ruiz, R., & Stützle, T. (2008). An iterated greedy heuristic for the sequence dependent setup times flowshop problem with makespan and weighted tardiness objectives. European Journal of Operational Research, 187(3), 1143-1159.
34. Ying, K. C. (2009). An iterated greedy heuristic for multistage hybrid flowshop scheduling problems with multiprocessor tasks. Journal of the Operational Research Society, 60(6), 810-817.

Dual tree wavelet transform based denoising of images using subband adaptive thresholding via genetic algorithm

A. Boukhobza [1] , A. Taleb-Ahmed[2], and A. Bounoua[3]

1. Departement of electronics, UHBC, Chlef, 02000, Algeria
aek_boukhobza@yahoo.ca

2. LAMIH UMR CNRS 8201 UVHC, Le mont Houy, 59313 Valenciennes Cedex 9, France
abdelmalik.taleb-ahmed@univ-valenciennes.fr

3. RCAM Laboratory, UDL, Sidi Bel-Abbes, 22000, Algeria
nacerbounoua@yahoo.fr

Abstract

In this paper, we propose an image denoising method based on 2D dual tree discrete wavelet transform with genetic algorithm optimization. The VisuShrink method has been selected for thresholding. Particularly in this work, the threshold value is corrected and optimized for each subband using a genetic algorithm. After the threshold is determined, a nonlinear threshold function is applied which make a good compromise between the soft threshold and hard threshold function. Experimental results indicate that the proposed method outperforms some of the well known noise removal operators in the literature in terms of PSNR (dB) and SSIM and obtain clearer image edges.

Keywords : image denoising; dual tree complex wavelet transform; wavelet thresholding; genetic algorithm.

1 Introduction

The discrete wavelet transform (DWT) has been recognized as an important tool for image denoising [1] [2] [3] [8]. This transform tool can effectively separate the noisy signal from the image signal. In wavelet domain, large coefficients correspond to the image, and small ones represent mostly noise. By filtering these coefficients with a suitable threshold, which is known as wavelet shrinkage, most of the noise power can be suppressed significantly while preserving main image features.

It is well known that 2D DWT is constructed using separable filter bank (i.e., rows and columns are processed separately) which is not enough effective to capture image features with arbitrary orientation such as edges and contours in images that are not aligned with the horizontal or vertical direction. For that reason, this transformation tool cannot provide efficient approximation for directional characteristics of images which in turn affects the performance of DWT-based denoising schemes. Many multiscale transforms have been derived from the DWT in order to improve directional selectivity, namely: curvelets, contourlets, wedgelet, complex wavelet transform [4]….

In this work, a new noise removal method based on DDWT is exposed. In this method, the threshold is adaptive to each subband of DDWT decomposition. The thresholding of wavelet coefficients in the transformed domain has been done using a weighted universal threshold (i.e., *VisuShrink*). The proposed algorithm searches the optimal weights adapted to each subband using a genetic algorithm. Experimental results show that the proposed method outperforms some of the well known noise removal operators in the literature in terms of PSNR (dB) and SSIM.

2 Dual-tree discrete wavelet transform

The dual tree complexe wavelet transform (DT-CWT) is one of many developed tools derived from DWT in order to get important additional properties, such as shift invariance and good directionality in two and higher dimensions.

DT-CWT is implemented using two real DWT trees; the first tree represents the real part of the transform while the second represents the imaginary part. The analysis part of the transform is illustrated in figure 1. Specially in this case, the filters used in the two trees of the DWT satisfy perfect reconstruction (PR) and should be jointly designed so that the overall transform is approximately analytic [4].

The form of the combined filters used in the DT-CWT 1D is given by:

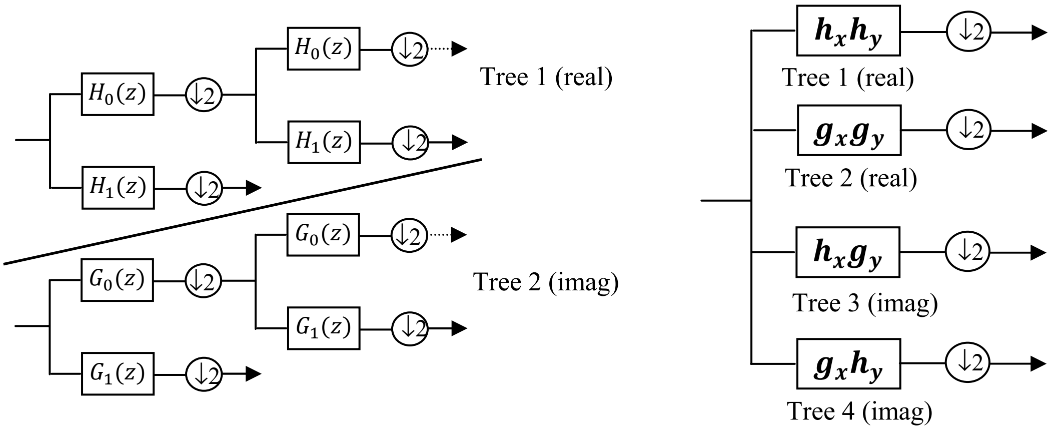
$$\mathrm{h_x} + j\mathrm{g_x} \tag{1}$$

Where $\boldsymbol{h_x}$ and $\boldsymbol{g_x}$ are respectively the sets of real-valued low-pass filters $\{h_0, h_1\}$ and high-pass filters $\{g_0, g_1\}$, in the direction x. Particularly in DT-CWT, filters applied in the first level of decomposition are numerically different than their respective at the other levels [4].

The 2D DT-CWT structure is an extension of conjugate filtering in 2D case. In this case, the pairs of filters are applied separately in the two dimensions (x and y), and this is expressed as:

$$(\mathrm{h_x} + \mathrm{jg_x})(\mathrm{h_y} + \mathrm{jg_y}) = (\mathrm{h_x h_y} - \mathrm{g_x g_y}) + j(\mathrm{h_x g_y} - \mathrm{g_x h_y}) \tag{2}$$

This results in four trees as shown in figure 2. Either the real part or the imaginary part of (2) can be used as a transform tool since they both satisfy perfect reconstruction and directional characteristic [4]. Real part of 2D DT-CWT is denoted DDWT and is used in our work in order to reduce the redundancy from 4:1 to 2:1.



(a) (b)

Figure 1: Analysis filter bank DT- DWT, (a) 1D, (b) 2D.

3 Wavelet thresholding

Wavelet thresholding is a widely used term for wavelet domain denoising, which means the procedure applied on wavelet transformed coefficients of the noisy image for noise suppression. It is applied only for wavelet coefficients of the detail subbands, while keeping the low resolution coefficients unaltered. Here, this procedure of thresholding is achieved with the following steps:

1. Decompose the noised image using 2D dual discrete wavelet transform (DDWT)
2. Select a given threshold value
3. Apply the threshold on the wavelet coefficients according to a shrinkage rule.
4. Get the denoised image by the inverse 2D dual trees discrete wavelet transform (IDDWT)

The shrinkage process is governed by the threshold value and the thresholding rule. For the latter, there are two frequently used methods: hard and soft thresholding. In hard thresholding, the wavelets insignificant (less than a predefined threshold) coefficients are set to zero and the significant coefficients remain unchanged. In soft thresholding, the absolute value of a significant coefficient is reduced by the threshold value. Hard thresholding suffers from abrupt discontinuity which causes artifacts in the denoised image while soft thresholding causes over smoothing in the denoised image.

In this work, a compromised thresholding method called Garrote thresholding [2] is employed. For a given threshold T, this function thresholding is defined by:

$$th(x,T) = \begin{cases} x - 0.5\frac{T^2}{x} & if\ |x| > T \\ 0.5\ \frac{x^3}{T^2} & if\ \ |x| \le T \end{cases} \qquad (3)$$

The key problem in the thresholding technique is the choice of the threshold. If this latter increase excessively, the noise will remain in the restored image. On the other hand, if this value is very small, some image details (edges, textures,…) would be lost. Figure 2 illustrates the three functions thresholding methods.

Thresholding methods can be classified into two categories, global thresholds and level dependent thresholds. In first class, a single threshold is applied globally to all subband coefficients, while in the second class a different threshold is estimated and applied for each subband.

The most known of global thresholds is the universal threshold (VisuShrink). It is a simple entropy measure totally dependent on the size of the signal. The threshold value is expressed as:

$$\lambda = \sigma\sqrt{2log\ (M)} \qquad (4)$$

Where M is the size of the signal and σ the noise standard deviation estimated from the subband by the robust median estimator:

$$\sigma_n = \frac{median(|y_{ij}|)}{0.6745} \qquad (5)$$

Where y_{ij} are the detail coefficients at the finest level of wavelet decomposition. The reason for this choice of level is that the corresponding wavelet coefficients are dominated by the noise.

One of subband adaptive thresholding efficient methods is BayesShrink, which is based on Stein's unbiased risk estimator [9]. In this case, the threshold is defined by:

$$\lambda = \frac{\sigma_n^2}{\sqrt{max(\sigma_y^2 - \sigma_n^2, 0)}} \qquad (6)$$

Where $\sigma_y^2 = \frac{1}{N}\sum_{i=1}^{N} y_i^2$ and N is the number of wavelet coefficients of subband.

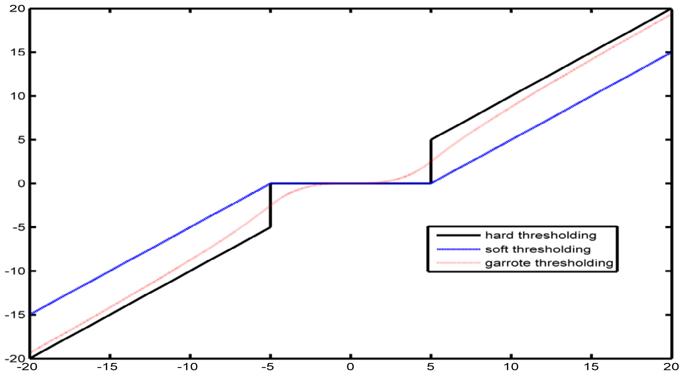


Figure 2: thresholding functions

3 GA based optimization

GA is a stochastic randomized search algorithm starting with an initial random population of problem solutions. It evolves iteratively through genetic operators: selection, crossover and mutation stages until obtaining a sufficient solution.

In experimentation, the population size is set to 40 and the initial population is obtained by randomly generating threshold correction weights that varies between 0.05 and 2. In order to prevent long chromosomes, a real chromosome is encoded to represent the values of these weights.

In experimentation, four levels of DDWT decomposition are used. In the implementation of DDWT, we have chosen the CDF 9/7 filter bank for the first stage and the 6-tap Qshift filters for the three remaining stages. The popular natural images Lena and Barbara are used in this work. These images are corrupted with Gaussian noise with different noise variances.

Particularly in this work, and in order to prevent long chromosomes and speed up the convergence of algorithm, GA is employed to search the optimal weights for each level independently of the other levels of DDWT trees (i.e., errors are independents). In this case, a chromosome of length six is employed to represent the weights of the subbands in a single level of DDWT structure. The maximum generation is set to 300 for the first level, 300 for the second, 200 for the third, and 100 for the last.

In genetic algorithm, a simulated binary crossover operator [6] with a distribution index of 20 with a probability of 0.98 is used, and a polynomial mutation of distribution index 20 with a probability of 0.05 is applied. Binary tournament selection has been used as genetic selection operator. In addition, a crowding distance has been employed to preserve population diversity during GA run.

In order to prevent loss of the best solutions obtained during GA run, the four (10% of population size) fittest chromosomes in the population of each generation are copied to the next generation created by crossover and mutation stages. The obtained population is sorted and truncated at the population size. This operation of genetic algorithm is known as Elitism. The flowchart of the proposed genetic algorithm is shown in Figure 3.

The objective of the optimization problem is to minimize the *MSE* (mean squared error) value between wavelets coefficients of noised image subbands and their corresponding of reference image subbands [7].

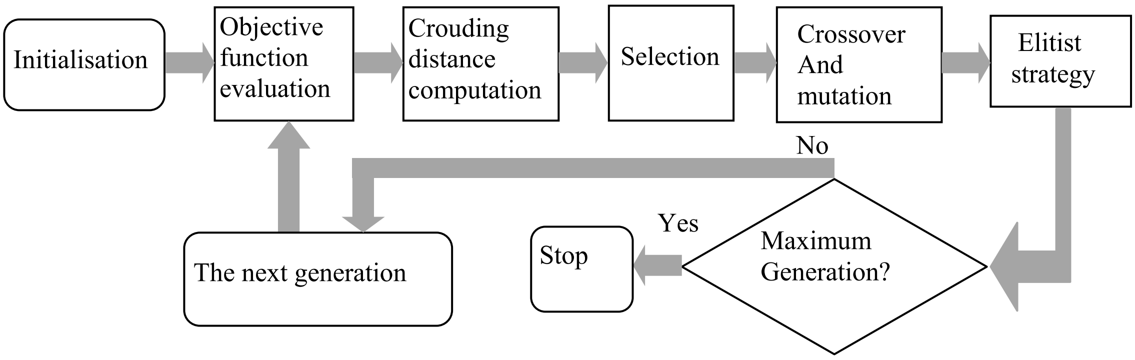


Figure 3: Flowchart of GA algorithm.

4 Simulation and results

To investigate the effectiveness of the proposed GA thresholding method, results obtained using the proposed based shrinking methods are compared with the DDWT with VisuShrink, DDWT with BayesShrink, and DWT with BayesShrink. For all these methods, we have chosen the best thresholding among those depicted in figure 2. The wavelet filter db4 is employed for the implementation of DWT.

Table 1 shows the PSNR and SSIM results of the denoising methods which we have just mentioned. The SSIM (Structural Similarity Index Measure) is a perceptual measure of quality of images [5]. From this table, it is clear that the proposed denoising method (DDWT-GA) performs better than the other methods. One can notice that PSNR gain of the proposed method with respect the other ones increases significantly with the variance.

For relatively smooth image Lena, we concluded from figure 5 that both low and high frequency components are better represented with our optimized thresholds.

Figure 6 shows a portion of Barbara image after noise reduction. From this figure, it is clear that the VisuShrink method smoothes out most image details and thus produces a blurred denoised image. It can be observed that textures lines are better preserved with the denoising methods based on DDWT (e and f) than the denoising with DWT (d). Through visual inspection of all denoised images, it can be observed that the proposed method DDWT-GA provides the best image quality.

In addition, for medical image ("med") exposed in figure 6, it is clear that this method outperforms all others methods and is well sufficient to suppress the Gaussian noise.

	DDWT & VisuShrink and garrote thresholding		DWT & bayesian soft thresholding		DDWT & bayesian soft thresholding		Proposed DDWT-GA	
Lena								
σ=10	32.22	0.865	33.33	0.862	34.00	0.888	34.83	0.898
σ=20	29.32	0.806	30.22	0.788	30.66	0.832	31.69	0.841
σ=30	27.64	0.760	28.50	0.743	28.70	0.788	29.86	0.800
Barbara								
σ=10	28.92	0.843	30.92	0.854	32.13	0.887	32.88	0.904
σ=20	25.52	0.733	27.16	0.742	27.52	0.789	28.76	0.811
σ=30	23.98	0.660	25.16	0.670	24.42	0.677	26.52	0.737
Med								
σ=20	35.01	0.926	35.29	0.858	35.49	0.902	36.93	0.934
σ=30	32.04	0.775	32.16	0.770	32.32	0.850	33.67	0.863
σ=50	29.86	0.814	30.49	0.720	30.14	0.748	31.78	0.820

Table 1: PSNR and SSIM (SSIM values in the right) results
for various denoising methods.

5 Conclusion

In this paper, we have presented a method based on DDWT for image denoising. Particularly in this work, denoising method using weighted universal threshold is applied. For a given corrupted image, the optimal weights adapted for each subband of DDWT are optimized using a genetic algorithm.

As it was seen in the simulations, results obtained demonstrate that the proposed method efficiently suppresses the Gaussian noise with different variances and produces better numerical results and visual quality.

References

[1] S. Mukhopadhyaya, J. K. Mandalb (*2013*), Wavelet based Denoising of Medical Images using Sub-band Adaptive Thresholding through Genetic Algorithm*.* Procedia Technology*,* 10*,* 680 – 689.

[2] M. Nasri, H. Nezamabadi-pour (2009). Image denoising in the wavelet domain using a new adaptive thresholding function, Neurocomputing, 72 1012–1025.

[3] F. Luisier, T. Blu, and M. Unser (2007). A New SURE Approach to Image Denoising: Inter-Scale Orthonormal Wavelet Thresholding. IEEE Transactions on Image Processing, 16(3), 593– 606.

[4] I. W. Selesnick, R. G. Baraniuk, and N. G. Kingsbury (2005). The dual-tree complex wavelet transform. IEEE Signal Process. *Mag.*, 22, 123–151.

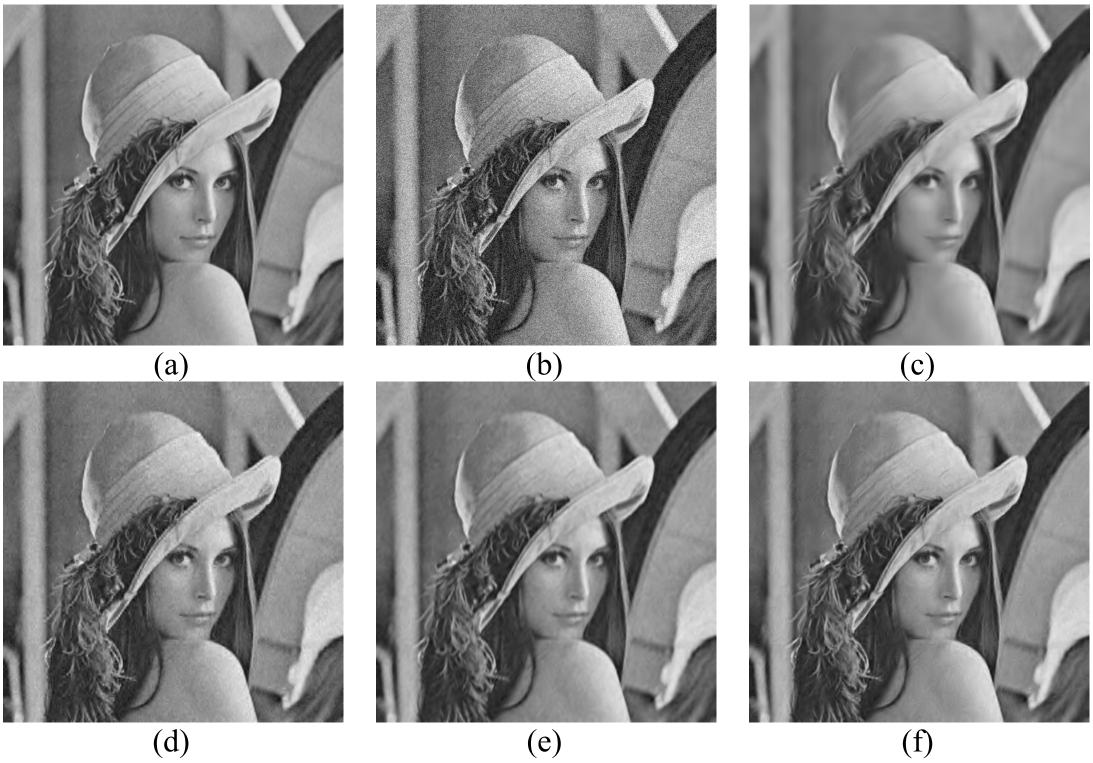
[5] Z. Wang, A. C. Bovik, H. R. Sheikh and E. P. Simoncelli (2004). Image quality assessment: From error visibility to structural similarity. IEEE Transactions on Image Processing. 13(4), pp. 600-612.

[6] M. M. Raghuwanshi, O. G. Kakde (2004). Survey on multiobjective evolutionary and real coded genetic algorithms. In Proc. of the 8th Asia Pacific Symposium on Intelligent and Evolutionary Systems, 150–161.

[7] X. P. Zhang (2001). Thresholding neural network for adaptive noise reduction. IEEE Trans. Neural Networks. 12(3), 567–584.

[8] C. Taswell (2000). The what, how and why the wavelet shrinkage. IEEE comput. sci.eng. 2(3), 12–19.

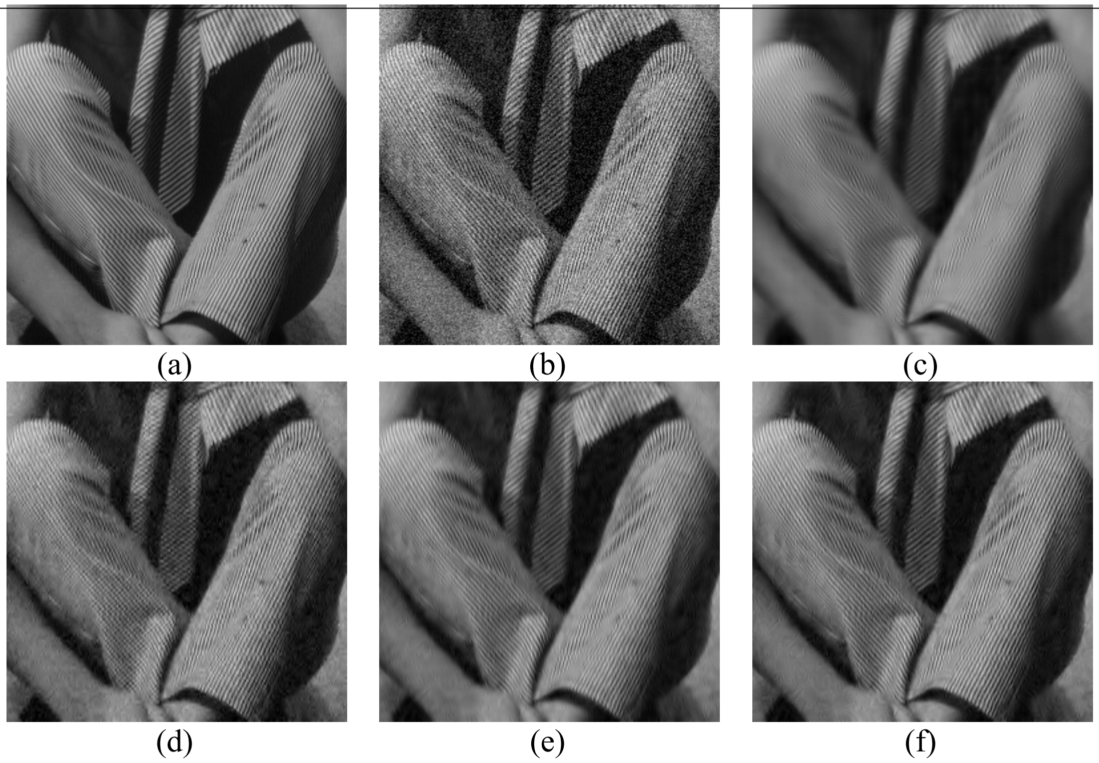
[9] S. Chang, B. Yu, M. Vetterli (2000). Adaptive wavelet thresholding for image denoising and compression. IEEE Trans. Image Process. 9, 1532–1546.



(a) (b) (c)

(d) (e) (f)

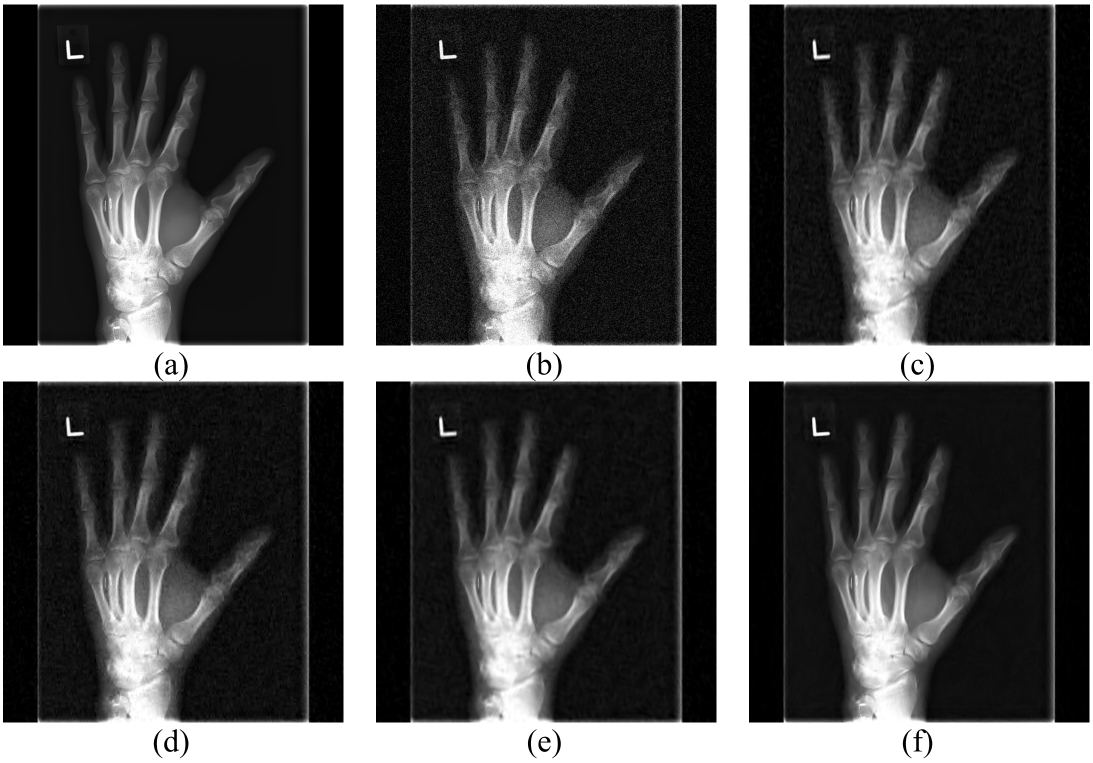
Figure 4: Comparing the performance of the various denoising methods on Lena image with σ=20. (a) original image, (b) noisy image, (c) DDWT&VisuShrink (d) DWT method, (e) DDWT & Bayesian thresholding, (f) DDWT-GA based method.



(a) (b) (c)

(d) (e) (f)

Figure 5: Comparing the performance of the various denoising methods on a portion of Barbara image with σ=20. (a) original image, (b) noisy image, (c) DDWT&VisuShrink (d) DWT method, (e) DDWT & Bayesian, (f) DDWT-GA based method.



(a) (b) (c)

(d) (e) (f)

Figure 6: Comparing the performance of the various denoising methods on medical image with σ=20. (a) original image, (b) noisy image, (c) DDWT&VisuShrink (d) DWT method, (e) DDWT & Bayesian, (f) DDWT-GA based method.

A Bi-Objective Maintenance-Routing Problem; an efficient solving approach

Mohammad Rahimi and El-Ghazali Talbi

INRIA Laboratory, CRISTAL-CNRS, 59650, Villeneuve d’Ascq, Lille, France
{mohammad.rahimi, el-ghazali.talbi}@inria.fr

1 Introduction

Regularly planned and scheduled maintenance is a critical requirement to reduce the occurrence of unforeseen failure and keeping the equipment running at peak efficiency. Maintenance scheduling becomes complex when the machines are geographically distributed. In this case, in addition to assigning the maintenance operations to technicians, it is needed to find the best set of routes for technicians’ visits. In fact, it is necessary to study the maintenance and the routing decisions simultaneously. Such a joint decision problem is known as the maintenance-routing problems.

To the best of our knowledge, few studies attempted to investigate the simultaneously maintenance scheduling and vehicle routing problem[1], [2]. For this purpose, in this study a new mathematical model is proposed to determine optimized maintenance-routing policy. Moreover, we are working to develop an efficient solving approach based on meta-heuristics to obtain the accurate and reliable results in a reasonable time.

2 Problem description

In this section a bi-objective mathematical model is proposed to determine optimized routing-maintenance policy, simultaneously. In this model, first objective function minimizes the total costs due to traveling, delay in start time of a Preventive Maintenance (PM)/Corrective Maintenance (CM) operation in customer while second objective function attempts to minimize the waiting times before beginning of the CM operations.

In this study there is a system with some customers geographically distributed, where each customer has one machine that should be visited and repaired by technician in different cycles. The PM operations are scheduled with a certain frequency to reduce the occurrence of unforeseen failure in the long term. Regarding the previous experiences, the time of unforeseen failure occurrence is known for each machine at each customer, but its repairing can be postponed until defined period. The time interval between occurrence of unforeseen failure and its repairing named waiting time. The set of technicians, who need to visit the set of machines to perform the PM/CM operations to prevent the system failure. The technician are different in duration time of doing a PM/CM operation which causes different in salary. A central depot is concerned as the point of departure and final destination. Since each technician should travel to perform PM/CM operation at the customer location, the distance between each two customer is defined. The main aim of this study consist of determining a joint routing-maintenance policy for all machines taking into account making a balance between the waiting time and total cost of system. In the optimized maintenance policy will be determined in which periods the PM and CM operations should be performed at each customer. Moreover, each technician will be assigned to customers and will be determined in which sequence should visit customers and perform PM and CM operations at each period.

The detailed conditions of system are summarized as follows:

- The technicians are different in skills regarding required time to perform a maintenance operation. The time required to perform a maintenance operation is depended on skill of assigned technician.

- The more skilled technician has more salary.
- All technicians are homogenous and are able to perform any PM/CM operation.
- The technicians start in the central depot in the beginning of each period and should return to the central depot by the end of the period.
- Each machine should be repaired by only one technician at each period. It means if the machine should be repaired in the specific period, only one technician should be assigned to the machine.
- The PM operation should be performed on the all the machines at the first period.
- In the case of no unforeseen failure occurs on the machine at planning horizon, the PM operations will be performed regarding the defined frequency. The frequency is defined regarding planning horizon and the duration of the interval between two consecutive PM operations.
- In the case of unforeseen failure occurrence on the machine, no predictive maintenance can be scheduled and performed before performing CM operation. In this case, CM operation should be scheduled to assign a technician on the machine until maximum L period. Moreover, next PM operation will be scheduled and performed after λ period.
- After performing a CM operation, the machine returns to the good condition and no unforeseen failure occurs until the next repairing that will be a PM operation in λ period. It means two unforeseen failure cannot be occurred consequently.
- The time required to perform a CM operation is longer than the time required to perform a PM operation on each machine.
- The CM cost is larger than the PM cost.
- The machines impose time windows to the system which means the technician should start maintenance operation before the latest possible start time. In cases where this time windows is not respected, a delay penalty applies if the technician starts after the latest allowed time.
- The travel time between two customers is depended to speed of vehicle in the rout at each period.

3 Mathematical Formulation

The objective function of the mathematical model associated with the presented framework is provided in this section. In the presented equations indexes *i* and *j* present the customers, *k* addresses technicians and *t* presents different periods. The parameters v_{ij}, r, p_i, c_k, pm_k and cm_k define distance between customers, transportation cost per unit distance, penalty cost of one unit time delay due to start time of a PM/CM operation at customer, one unit time cost of a PM/CM operation, time required to perform a PM operation and time required to perform a CM operation, respectively. Moreover, x_{ijkt}, d_{it}, a_{ikt}, b_{ikt} and y_{itt} are decision variables which present customer visiting, amount of delay, performing PM/CM operation and delay occurred in visiting customer to perform a CM operation.

$$Min \quad f_1 = \sum_{i,j,k,t} x_{ijkt}.v_{ij}.r + \sum_{i,t} d_{it}.p_i + \sum_{i,k,t} a_{ikt}.c_k.pm_k + \sum_{i,k,t',t} b_{ikt}.c_k.cm_k \tag{1}$$

$$Min \quad f_2 = \sum_{i,t} y_{itt} \tag{2}$$

The first objective function (1) minimizes the total cost which consist of traveling cost between customers, penalty cost due to start time out of time windows and the wages of technicians for PM/CM operations. The second objective function (2) optimizes the customer satisfaction level by minimizing the waiting times until performing a CM operation in the case where an unforeseen failure occurs.

4 Resolution approach

To proof the feasibility of proposed bi-objective mathematical model, we use the “GAMS v22.2” optimization software using solver CPLEX v10.1.

The proposed model is known as a NP-hard problem [3] which causes we cannot solve it in a reasonable time for the big sizes. For this purpose, we are working to develop an efficient solving approach based on hybridizing two meta-heuristics. Then, we will show the efficiency of proposed approach using the defined measurement.

References

[1] E. López-Santana, R. Akhavan-Tabatabaei, L. Dieulle, N. Labadie, and A. L. Medaglia, “On the combined maintenance and routing optimization problem,” *Reliab. Eng. Syst. Saf.*, vol. 145, pp. 199–214, Jan. 2016.

[2] R. Macedo, R. Benmansour, A. Artiba, N. Mladenović, and D. Urošević, “Scheduling preventive railway maintenance activities with resource constraints,” *Electron. Notes Discret. Math.*, vol. 58, pp. 215–222, Apr. 2017.

[3] M. Rahimi, A. Baboli, and Y. Rekik, “Multi-objective inventory routing problem: A stochastic model to consider profit, service level and green criteria,” *Transp. Res. Part E Logist. Transp. Rev.*, vol. 101, pp. 59–83, 2017.

Heat exchanger network synthesis with an enhanced superstructure and hybrid metaheuristics

Leandro V. Pavão[1], Caliane B. B. Costa[1] and Mauro A. S. S. Ravagnani[1]

1. Department of Chemical Engineering, State University of Maringá, Av. Colombo, 5790, Bloco D90, CEP 87020900, Maringá, PR, Brazil
leandropavao@gmail.com; cbbcosta@uem.br; massravagnani@uem.br

Abstract. Superstructure-based optimization models are widely used for the synthesis of heat exchanger networks (HEN). Near-optimal sets of heat exchangers and piping configurations may be achieved when those models are solved, although this is not a trivial task given the models' complexity. In this work, a new superstructure-based model is presented for HEN synthesis containing new options for the allocation of heaters and coolers. Those units are commonly placed at the end of process streams in order to provide/remove the required/exceeding heat that was not exchanged among process streams. Such new options make the model more complex to solve, with a larger number of possible structural combinations. SA-RFO, which stands for Simulated Annealing – Rocket Fireworks Optimization, is the hybrid meta-heuristic solution approach applied to solve the model. Such method encompasses features of SA, a continuous adaptation of SA, and Particle Swarm Optimization (PSO). The methodology is applied to a large-scale case study under two different scenarios (single and multiple types of utilities). Results show that HEN total annual costs can be reduced by placing some of those devices before other heat exchangers, outperforming previously reported HEN synthesis methodologies.

Keywords : optimization, heat exchanger networks, metaheuristics, mathematical modelling, process synthesis

1 Introduction

Several HEN synthesis methodologies are based on the solution of mathematical programming models derived from superstructures for total annual costs (TAC) minimization. A prominent concept on the field is the stage-wise superstructure (SWS) proposed by Yee and Grossmann [1]. Relatively simple, that model has been taken as basis for the development of several subsequent HEN synthesis works. HEN synthesis models for costs optimization are typically mixed-integer nonlinear programming (MINLP) problems, which require efficient solution methodologies for attaining near-optimal solutions. Pursuing HEN optimality, meta-heuristics are a sort of solution strategy that has reached promising results. These rely on random searches and specific heuristic rules when seeking for optima. Some examples of HEN synthesis contributions using the SWS under the meta-heuristics scope employed techniques such as Particle Swarm Optimization [2], hybrid GA with PSO and parallel processing techniques [3] and Simulated Annealing (SA) with Rocket Fireworks Optimization [4]. Moreover, besides costs, other aspects have been considered in HEN synthesis, such as environmental performance [5] and financial risks [6].

Despite its wide use, the SWS of Yee and Grossmann [1] has some drawbacks. For instance, heaters and coolers can be placed only at streams ends. However, solutions with auxiliary heating/cooling in intermediate stages (i.e., before heat exchange among process streams) could be promising and lead to better objective functions values. This work proposes a new superstructure based on the SWS where utilities can be placed at extra stream split branches in all stages. Ending stages are maintained for exhausting/providing energy for achieving streams' target temperatures.

The solution strategy used to solve the proposed model is the SA-RFO method [4], which had to be re-worked in order to include the routines handling the new variables.

2 Mathematical model

The HEN synthesis mathematical model here used is an extension of the well-used SWS model by Yee and Grossmann [1]. The original SWS introduced the “stages” concept. In each stage of the SWS, all possible streams pairing for heat exchange were possible. Each heat exchanger was placed at a stream split branch in a stage. That configuration implied that no sequential heat exchanger chains were possible at a stream in a given stage. Heaters and coolers were placed at streams ends for matching their energetic needs not fulfilled by means of heat exchange in stages of the HEN structure. Aiming for a broader search, the present proposal is to include extra branches in all stages for enabling the use of utilities not only at streams ends. Note that more stream split branches might be required if multiple types of utilities are available. Figure 1 presents an illustration of the new SWS with one type of hot utility and one type of cold utility.

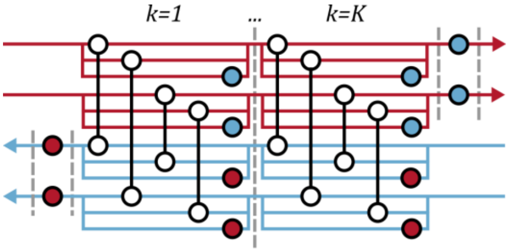


Figure 1. Enhanced stage-wise superstructure

In the model arisen from the new superstructure, HEN operating costs, i.e., those related to utilities requirement, are as follows.

$$
\begin{aligned}
OC = \ & \sum_{i}\sum_{n} Ccu_{n} \cdot FhQcuEnd_{i,n} \cdot QcuEnd_{i,n} + \\
& \sum_{i}\sum_{n}\sum_{k} Ccu_{n} \cdot QcuInt_{i,n,k} + \\
& \sum_{m}\sum_{j} Chu_{m} \cdot FcQhuEnd_{m,j} \cdot QhuEnd_{m,j} + \\
& \sum_{m}\sum_{j}\sum_{k} Chu_{m} \cdot QhuInt_{m,j,k} + \\
& i \in N_{H}, m \in N_{HU}, j \in N_{C}, n \in N_{CU}, k \in N_{S}
\end{aligned}
\tag{1}
$$

where *m* and *n* indexes represent the utility number (some plants may have more than one type). The *cu* and *hu* suffixes are for cold and hot utilities, respectively. *Ccu* and *Chu* are utility costs, *QhuEnd* and *QcuEnd* are the required/exceeding heat for a stream which was not provided/exhausted in stages. *FhQcuEnd* and *FcQhuEnd* are the fraction exhausted/provided by means of a given utility at streams ends (note that splits might be present at streams ends as well in case of multiple utilities). *QcuInt* and *QhuInt* are heat loads of heaters/coolers placed in the stages of the superstructure.

HEN capital costs are obtained as follows.

$$
\begin{aligned}
CC = \ & \sum_{i}\sum_{j}\sum_{k} z_{i,j,k} \cdot (B + C \cdot A_{i,j,k}{}^{\beta}) + \\
& \sum_{i}\sum_{n}\sum_{k} zcuInt_{i,n,k} \cdot (B + C \cdot AcuInt_{i,n,k}{}^{\beta}) + \\
& \sum_{m}\sum_{j}\sum_{k} zhuInt_{m,j,k} \cdot (B + C \cdot AhuInt_{m,j,k}{}^{\beta}) + \\
& \sum_{i}\sum_{n} zcuOut_{i,n} \cdot (B + C \cdot AcuOut_{i,n}{}^{\beta}) + \\
& \sum_{m}\sum_{j} zhuOut_{m,j} \cdot (B + C \cdot AhuOut_{m,j}{}^{\beta}) + \\
& i \in N_{H}, m \in N_{HU}, j \in N_{C}, n \in N_{CU}, k \in N_{S}
\end{aligned}
\tag{2}
$$

where *B*, *C* and *β* are cost related parameters. Variables beginning with "*A*" are related to areas, and their suffixes are analogous to those presented in Eq. (1). All "*z*" variables are binary and represent a unit existence/absence.

Finally, the HEN optimization model is written as follows.

$$\begin{aligned} (HEN\_OPT) \quad & \min \quad \{TAC = OC + CC\} \\ & s.t. \quad HEN \quad constraints \end{aligned} \tag{3}$$

HEN constraints regard thermodynamic feasibility conditions, heat balances, temperatures, logarithmic mean temperature differences and area calculations, which lead to the final TAC of a HEN solution. It is worth mentioning that in the model derived by Yee and Grossmann [1], an isothermal mixing simplification was assumed, avoiding the need for calculating heat balances in the mixers. Such assumption is not here made and those balances are always performed.

3 Metaheuristic solution approach

The strategy here used for achieving minimal costs solutions is a hybrid meta-heuristic based on Simulated Annealing (SA) and the Rocket Fireworks Optimization (RFO) [4]. SA-RFO approach is a two-level HEN optimization strategy. That is, the method entails an upper level combinatorial optimization method (Simulated Annealing [7]) that handles HEN topology, while the lower level method (RFO) handles the continuous domain. Considering that the method starts with an "empty" topology (*i.e.*, no units exchanging heat between process streams), it can be briefly described as follows:

i) a random heat exchanger, heater or cooler is added to the topology;

ii) Rocket Fireworks Optimization is applied to the new topology to find heat loads and stream split fractions that lead to optimal costs;

iii) the optimal solution to that topology is returned to the upper level;

iv) the new topology is accepted or discarded according to Simulated Annealing rules, according to the costs returned by RFO;

v) if the new solution is the best one found so far, it is recorded;

vi) SA termination rules are checked. If the criteria are met, the procedure is terminated and the best solution returned. Otherwise, the procedure goes back to (i).

SA rules mentioned in step (iv) accept the new topology if RFO returns a price to that given topology lower than that associated to the current topology or, in case that cost is greater, if the following equation is satisfied:

$$rand(0,1) < \exp\left(\frac{TAC^{New} - TAC^{Current}}{T}\right) \tag{4}$$

where rand(0,1) is a random number generated from a uniform distribution between zero and one, TAC^{New} and $TAC^{Current}$ are the optimal TACs of new and current topologies and *T* is the "temperature", a parameter of SA. Note that, for higher values of *T*, there is a higher probability of acceptance for topologies with greater costs than that of the current topology. Such measure aids the method in avoiding local minima premature stagnation.

It is worth mentioning that RFO is a combination of two continuous optimization approaches. First, a continuous adaptation of SA (CSA) is conducted. The CSA procedure is similar to the common SA, but with random continuous moves performed to the continuous variables. The best solution found by CSA is included to a swarm of random solutions. This swarm is subsequently evolved by applying Particle Swarm Optimization [8]. Figure 2(a) presents an illustration of the method behavior, evidencing the similarity to an ascending rocket firework and its explosion. Figure 2 (b) shows the application of RFO to a topology in a relatively simple four-stream problem [9]. It is possible to note that CSA quickly evolves towards a minimum region (only accepted solutions were plotted), and to observe the generated particles heading towards that region. Note also the presence of penalized solutions. Those are thermodynamically infeasible (hot stream outlet temperature of the (1,2,1) heat exchanger is lower than the cold stream inlet temperature). Their monetary value was capped to $20,000 for better visualization in the plot but were actually higher and variable. The penalty function, which is directly summed to the TAC if a constraint is unsatisfied, is defined generally as follows:

$$pen = P1 + P2 \cdot (X^{Bound} - X)^2 \quad (5)$$

where *P1* and *P2* are penalty constants, and *X* and X^{Bound} are a generic variable and its violated bound, respectively.

SA-RFO method was adapted to include moves in SA related to the addition of heaters and coolers in stages. Alterations were performed in the lower level as well for managing new heat loads and stream split fractions associated to the stage new units.

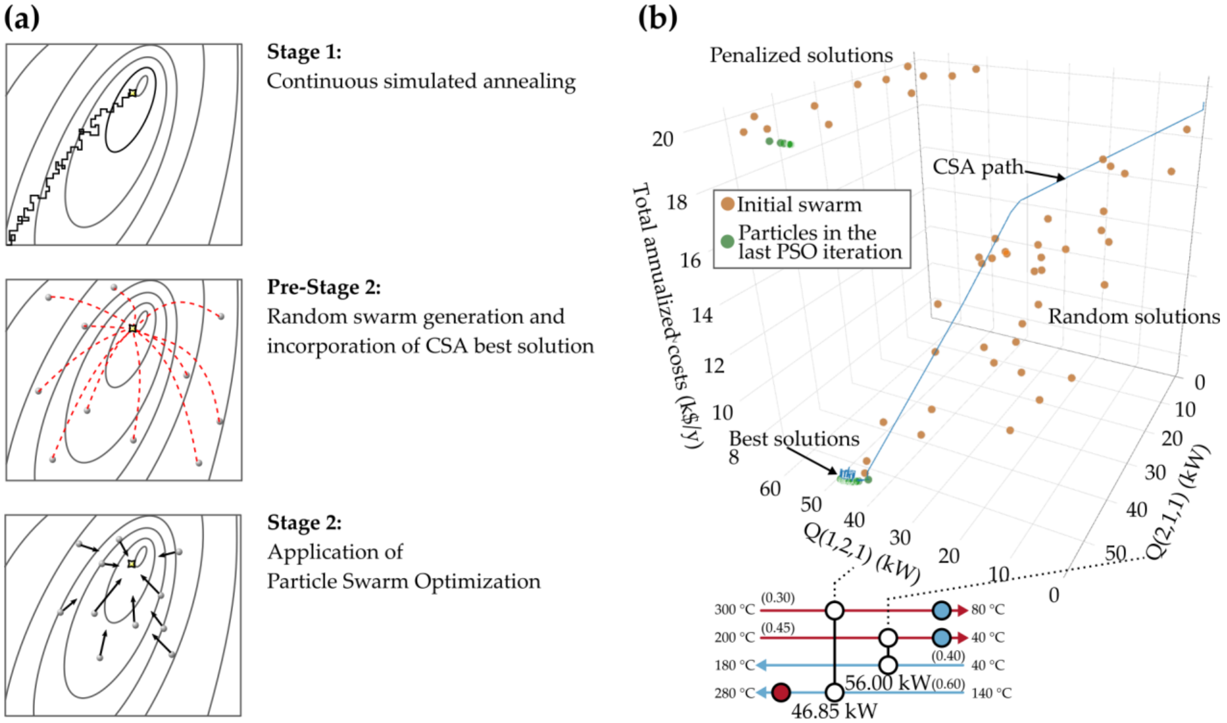


Figure 2. Illustration of the stages of Rocket Fireworks Optimization and application to a topology: (a) illustration of RFO stages; (b) 3-D representation of RFO application to a simple topology;

4 Case study

In order to illustrate the presented methodology potentialities, it is here applied to an industrial scale case study. This example was introduced by Soršak and Kravanja [10], who considered different types of heat exchangers. Here, Escobar and Trierweiler's [11] adaptation is used (problem data can be found therein), which considers shell and tube exchangers and was used as a benchmark example in other works in the literature. The solution obtained by Escobar and Trierweiler [11] had TAC of 1,461,006 $/y. The authors applied the DICOPT solver in GAMS to solve the problem with the SWS model of Yee and Grossmann (1990), including the isothermal mixing assumption. However, some inconsistencies in that solution were found by Pavão et al. [3], who also revised it, finding TAC of 1,537,086 $/y. In that work, Escobar and Trierweiler's [11] solution was used as an initialization solution for their PSO method, which was improved to TAC of 1,516,482 $/y. Xiao et al. [12] employed a Random Walk with Compulsive Evolution (RWCE) meta-heuristic strategy to a no-splits formulation, and were able to find TAC of 1,509,749 $/yr. Bao et al. [13] combined the RWCE to an optimum protection strategy and identified the best solution so far, with TAC of 1,462,323 $/y. The present approach was able to outperform previous literature solutions, attaining

TAC of 1,433,419 $/y. Table 1 presents a comparison to other solutions reported in the literature regarding, besides TAC, some of their design aspects.

Table 1. Case study results comparison

Ref.	TAC ($/y)	Units	Qhu (MW)	Qcu (MW)	Area (m^2)
Escobar and Trierweiler [11]	1,537,086	21	1938.0	106.9	5551.1
Pavão et al. [3]	1,516,482	21	1938.0	106.9	5389.0
Xiao et al. [12]	1,509,749	23	1867.7	36.6	Not reported
Bao et al. [13]	1,462,323	22	2077.5	250.4	5053.2
This work	1,433,419	22	2495.2	664.1	4688.8

Apart from the original case study conditions, this problem was here investigated including multiple utility options. Table 2 presents data of the utilities that were included. High and medium pressure steam were added as hot utilities (labelled as HU2 and HU3), and air-cooling was included as cold utility (labelled as CU2). Figure 3 presents the solutions attained to both Scenarios 1 (as taken from Escobar and Trierweiler, [11]) and 2 (with additional utilities). The TAC obtained with the latter has TAC of 1,304,515 $/yr. Evidently, this result is not comparable to the literature given that, with the inclusion of new utilities, this scenario can be considered as a new problem, whose options were not available to previous authors. However, it is interesting to note that the present methodology was able to perform a cost-efficient utility selection.

Regarding structural advantages, note, in Scenario 1, that a heater is placed prior to a process stream to process stream heat exchanger at cold stream 7. If that heater were placed at that stream's end, TAC would increase by 58,676 $/y (4.1%).

Table 2. New hypothetical utilities

	Inlet temperature (°C)	Outlet temperature (°C)	Heat transfer coefficient (kW/m^2 °C)	Costs ($/kW)
Hot Utility 2	250	250	3.0	150
Hot Utility 3	200	200	3.0	100
Cold Utility 2	20	30	0.7	15

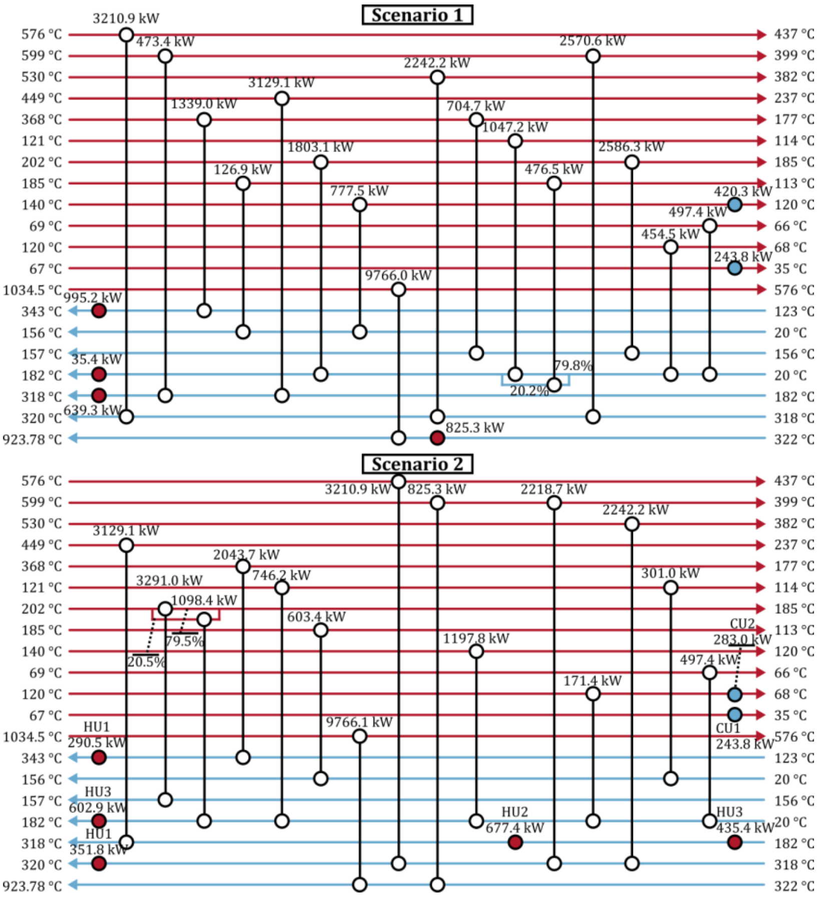


Figure 3. Optimal HEN for Scenarios 1 and 2

In Scenario 2, a heater is placed before another one at cold stream 5. HU3 (200 °C) raises the stream temperature to nearly 196 °C before HU2 (250 °C) leads it to 217 °C. In the sequence, that stream receives heat from hot stream 4 in a heat exchanger. If a heater was allowed only at the end of the stream, HU1 would be the only thermodynamically feasible utility to be used, and, in that case, TAC would raise by 66,099 $/y (5.1%).

5 Conclusions

A new extended stage-wise superstructure for HEN synthesis was developed, giving rise to a MINLP mathematical model. A meta-heuristic approach was revamped to handle the new complexities of the formulation. The methodology was applied to a large-scale problem in two scenarios: with one hot and one cold utility, as in the literature, and with multiple utilities. For Scenario 1, a solution with costs 5.1% lower than the best previously reported was found. In Scenario 2, an interesting solution comprising optimal choice of utilities was achieved using the newly developed aspects of the present model. It could be observed that allocating utilities to positions preceding heat exchange among process streams can lead to promising gains, outperforming previous methodologies.

6 Acknowledgements

The authors acknowledge the financial support from CAPES (Coordination for the Improvement of Higher Education Personnel, Brazil) and CNPq (National Council for Scientific and Technological Development, Brazil).

References

[1] T.F. Yee, I.E. Grossmann, Simultaneous optimization models for heat integration—II. Heat exchanger network synthesis, Comput. Chem. Eng. 14 (1990) 1165–1184. doi:10.1016/0098-1354(90)85010-8.

[2] A.P. Silva, M.A.S.S. Ravagnani, E.C. Biscaia, J.A. Caballero, Optimal heat exchanger network synthesis using particle swarm optimization, Optim. Eng. 11 (2010) 459–470. doi:10.1007/s11081-009-9089-z.

[3] L.V. Pavão, C.B.B. Costa, M.A.S.S. Ravagnani, Automated heat exchanger network synthesis by using hybrid natural algorithms and parallel processing, Comput. Chem. Eng. 94 (2016) 370–386. doi:10.1016/j.compchemeng.2016.08.009.

[4] L.V. Pavão, C.B.B. Costa, M.A.S.S. Ravagnani, L. Jiménez, Large-scale heat exchanger networks synthesis using simulated annealing and the novel rocket fireworks optimization, AIChE J. 63 (2017) 1582–1601. doi:10.1002/aic.15524.

[5] L. V Pavão, C.B.B. Costa, M.A.S.S. Ravagnani, L. Jiménez, Costs and environmental impacts multi-objective heat exchanger networks synthesis using a meta-heuristic approach, Appl. Energy. (2017). doi:10.1016/j.apenergy.2017.06.015.

[6] L. V. Pavão, C. Pozo, C.B.B. Costa, M.A.S.S. Ravagnani, L. Jiménez, Financial risks management of heat exchanger networks under uncertain utility costs via multi-objective optimization, Energy. 139 (2017) 98–117. doi:10.1016/j.energy.2017.07.153.

[7] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, Optimization by simulated annealing., Science. 220 (1983) 671–80. doi:10.1126/science.220.4598.671.

[8] J. Kennedy, R. Eberhart, Particle swarm optimization, in: IEEE Int. Conf. Neural Networks, 1995: pp. 1942–1948. doi:10.1109/ICNN.1995.488968.

[9] S. Ahmad, Heat Exchanger Networks, Cost Tradeoffs in Energy and Capital, Ph.D. Thesis, UMIST, Manchester, UK, 1985.

[10] A. Soršak, Z. Kravanja, Simultaneous MINLP synthesis of heat exchanger networks comprising different exchanger types, Comput. Chem. Eng. 26 (2002) 599–615. doi:10.1016/S0098-1354(01)00779-7.

[11] M. Escobar, J.O. Trierweiler, Optimal heat exchanger network synthesis: A case study comparison, Appl. Therm. Eng. 51 (2013) 801–826. doi:10.1016/j.applthermaleng.2012.10.022.

[12] Y. Xiao, G. Cui, T. Sun, J. Chen, An integrated random walk algorithm with compulsive evolution and fine-search strategy for heat exchanger network synthesis, Appl. Therm. Eng. 128 (2018) 861–876. doi:10.1016/J.APPLTHERMALENG.2017.09.075.

[13] Z. Bao, G. Cui, J. Chen, T. Sun, Y. Xiao, A novel random walk algorithm with compulsive evolution combined with an optimum-protection strategy for heat exchanger network synthesis, Energy. (2018). doi:10.1016/j.energy.2018.03.170.

Transformer's Health Index using Computational Intelligence

Ramon Alves dos Santos[1], Nadia Nedjah[1], Luiza de Macedo Mourelle[1] and Leonardo Torres Bispo dos Santos[2]

[1] Postgraduate Program in Electronics Engineering, State University of Rio de Janeiro,
Rio de Janeiro, Brazil
ramon.alves.rj@gmail.com, nadia@eng.uerj.br, ldmm@eng.uerj.br
[2] Research Center of Electric Energy, Rio de Janeiro, Brazil
ltorres@cepel.br

1 Introduction

Power transformers are equipments installed in high power substations with the highest investment, corresponding to about 60% of the invested capital [1]. The Brazilian power system, according to the Sistema Interligado Nacional (SIN - National Interconnect System) performance indicators provided by Operador Nacional do Sistema (ONS - System National Operator) in the years of 2012 and 2015, shows an upward increase of the indicator of disturbances in the basic network, having a reduction of 1.57% in the year 2016, but still maintaining a total number of 3201 disturbances [2].

The failure of high-powered assets can be disastrous and results in direct and indirect costs for the industrial, commercial, and residential sectors [3], besides having a negative effect on the social life of users of electric power [4]. In this way, a disruption in the supply of electric energy, sometimes cannot have its losses fully measured, being able to surpass the value of investment in the asset. Thereby, the evaluation of the operational state of power transformers is an essential stage for maintenance of the electric power supply and increase in reliability of the network.

In the last years, the priorization of assets became a key issue, due to the scarcity of resources and a difficulty of disconnection of assets. Aiming to verify the operational state of high power assets, several equipments, procedures and methodologies arouse. However, with a large amount of equipment evaluation methods, sometimes contradictions between results of different tests happen, not being trivial a composition among them in order to obtain the operational condition of the equipment. In this scenario, the use of methodologies based on the Health Index provides the means to overcome these problems.

Health Index is a methodology that seeks the composition of several tests, necessary to verify the operational condition of the transformer, aiming at a numerical index, which will represent the condition of the asset and allow to establish a prioritization. The composition of the index is traditionally dependent on experts, who assign scores for results from electrical, chemical and physical equipment test. For each factor evaluated, they establish weights that ponder the results, composing, by means of a summation, a numerical value, which will indicate the operational state of the equipment. This value is then used for the ranking of the set of assets in a substation. However, there is no consensus, among authors, on the composition of weights, falling into certain randomness. This limits the composition of the Health Index with a comprehensive approach, in which factors, such as the power of an equipment family, do not interfere in the composition.

The importance of having the means to compose several tests lead us to establish the operational condition of the equipment and, therefore, establish a prioritization among a set of assets, which do not always share the same characteristics in a substation. This is an opportunity to develop of a tool with a more generic approach. In this work, we present a new proposal for the classification and prioritization of high power transformers, performing the composition of the Health Index with computational intelligence techniques, such as Particle Swarm Intelligence, Elephant Herding Optimization and Genetic Algorithm.

In Section 2, the main methods, used in this project for the solution development and performance comparison, are presented. In Section 3, the evaluated features, for composition of the index, are described. In Section 4, the proposed method, for the prioritization and ranking problems, is introduced. In Section 5, the results achieved are presented. In Section 6, the conclusions are drawn together with future work.

2 Computational Intelligence Techniques

Computational intelligence techniques have been widely used to solve a diversity of engineering problems in recent years. In this section, we introduce the main techniques used in this work, in order to propose a new method for the classification and prioritization of high power transformers.

2.1 Particle Swarm Optimization

Particle Swarm Optimization (PSO) is a method developed by Kennedy and Eberhart [5], used for solving NP-hard problems, where no adequate solution is found with traditional methods, not requiring large specifications of the problem. In this method, each particle of the swarm represents a possible solution of the problem, having the strategy of moving the particles in a n-dimensional search space, based on the social behavior of a flock of birds or school of fish. The selection of the best solution is performed by a fitness function, which must be modeled for each problem, so that the particle, which presents a good solution to the physical problem, receives a good evaluation in the algorithm. At each iteration, each particle is influenced by the process of cognitive and social learning. The process directs the particles to promising regions until a stopping criterion is reached, such as a maximum number of iterations ($nMaxIt$) or a given value of fitness of the solution.

The main steps of the PSO algorithm are described in algorithm 1. In this algorithm, each particle has a velocity and an adaptive direction [5] that determine its next movement within the search space. The particle is also endowed with a memory that makes it able to remember the best previous position it passed by. In Algorithm 1, we denote by $Pbest[i]$ the best fitness particle i has achived so far and $Pbestx[i]$ the coordinates of the position that yield it. In the same way, we denote $Lbest[i]$ the best local fitness particle i and its neighbors have achieved so far and $Lbestx[i]$ the coordinates of the position that yield it.

Algorithm 1 Local Best PSO

```
for $i = 1$ $to$ $n$ do
  initialize the information of particle $i$
  initialize the position and velocity of particle $i$
end for
repeat
  for $i = 1 \to n$ do
    compute $fitness[i]$
    if $fitness[i] \leq Pbest[i]$ then
      update $Pbestx[i]$ using position of particle $i$
    end if
    if $Pbest[i] \leq Lbest[i]$ then
      update $Lbestx[i]$ using the $Pbestx[i]$
    end if
  end for
  for $i = 1$ $to$ $n$ do
    update velocity and position of particle $i$
  end for
until $nMaxIt$
return $Lbest[i]$ and $Lbestx[i]$
```

The social component, denominated Local Best, reflects information exchanged within the neighborhood of the particle [6]. The Local Best PSO is less susceptible to being trapped in local minimum and also the the ring topology used improves performance. The velocity is the element that promotes the capacity of particle locomotion and can be computed as described in equation

1 [5] [6], wherein w is called *inertia weight*, r_1 and r_2 are random numbers in [0,1], c_1 and c_2 are positive constants called *cognitve* and *social* terms, respectively, y_{ij} is the personal best position *pbest* found by the particle i so far, w.r.t. dimension j, and l_{ij} is the local best position *lbest* found by the particle i, w.r.t. dimension j, in the neighborhood. The position of each particle is updated as described in equation 2. Note that $x_{i,j}^{(t+1)}$ is the current position and $x_{i,j}^{(t)}$ is the previous position:

$$v_{i,j}^{(t+1)} = wv_{i,j}^{(t)} + c_1 r_1 \left(y_{i,j} - x_{i,j}^{(t)} \right) + c_2 r_2 \left(l_{i,j} - x_{i,j}^{(t)} \right), \tag{1}$$

$$x_{i,j}^{(t+1)} = v_{i,j}^{(t+1)} + x_{i,j}^{(t)}. \tag{2}$$

The velocity component drives the optimization process, reflecting both the experience of the particle and the exchange of information between the particles. The particle's experimental knowledge is referred to as the cognitive behavior, which is proportional to the distance between the particle and its best position found, w.r.t. its first iteration. The maximum velocity $v_{k,max}$ is defined for each dimension k of the search space. It can be expressed as a percentage of this space, according to equation 3, wherein $x_{k,max}$ and $x_{k,min}$ are the maximum and minimum limits of the search space explored, w.r.t. dimension k, respectively and $\delta \in [0,1]$:

$$v_{k,max} = \delta(x_{k,max} - x_{k,min}). \tag{3}$$

In the context of our work, we defined $w = 1$, $c1 = 1.5$, $c2 = 2$, $v_{max} = 0.5$, $x_{max} = 1$, $x_{min} = 0$, $n = 10$, 20, 50 and 100, and $nMaxIt = 100$ and 1000.

2.2 Elephant Herding Optimization

Elephant herd optimization (EHO) is a recent swarm intelligence method presented in 2015 [7]. The method has its proposal focused on the search for global solutions in n-dimensional search spaces, with its implementation inspired by the social behavior of Asian and African elephants. In nature, the elephants of the herd move under the matriarch's guidance and the male elephants, entering into adulthood, abandon the group. To represent this behavior artificially, two main operators are proposed in the method: an aggregation operator and a separation operator. In the artificial herd, the best solution in each generation is represented by the matriarch and the worst solution by the adult male elephant. Thus, with the aggregation operator, the elephant position of the herd is updated under the influence of the matriarch and, with the separation operator, the replacement of the worst solution of the artificial elephants is realized.

Each elephant has its position $x_{new,c_i,j}$ in the clan c_i updated by the operator described by equation 4:

$$x_{new,c_i,j} = x_{c_i,j} + \alpha \cdot (x_{best,c_i} - x_{c_i,j}) \cdot r, \tag{4}$$

wherein x_{best,c_i} is the matriarch of clan i, $x_{c_i,j}$ is the actual position of elephant j, $\alpha \in [0,1]$ means the matriarch's influence inside the clan and $r \in [0,1]$ is an uniform distributed random number. However, the movement of matriarchs is modeled by equation 5:

$$x_{new,c_i,j} = \beta \cdot x_{center,c_i}, \tag{5}$$

in which x_{center,c_i} is the center of clan c_i defined by the average coordinates of its elephants, $\beta \in [0,1]$ represents the influence of x_{center,c_i}. Algorithm 2 describes the agregation operator of the EHO method applied in this proposal.

The separation operator models when male elephants reach puberty and leave their clans. For each clan, the worse elephant is abandoned, and a substitute is created randomly inside the limits of the clan [7]. The modification of EHO proposed in this work is to place the substitute close to the matriarch, instead of restricting its position in the area occupied by the clan, as shown in equation 6:

$$x_{worst,c_i} = x_{best,c_i} + rand, \tag{6}$$

wherein x_{worst,c_i} is the worst of all elephants of the clan c_i, x_{best,c_i} is the matriarch of clan c_i and $rand \in [0,1]$ is an uniform distributed random number. Algorithm 3 describes the separation operator of the EHO method applied in this proposal.

Algorithm 2 EHO agregation operator algorithm

```
for $c_i$=1, ...,CLAN do
    for j=1,...,$n_{c_i}$ do
        Update $x_{c_{i,j}}$
        Generate $x_{novo,c_{i,j}}$ by Equation 4
        if $x_{c_{i,j}}$=$x_{melhor,c_i}$ then
            Update $x_{c_{i,j}}$ and generate $x_{novo,c_{i,j}}$
            Generate $x_{novo,c_{i,j}}$ by Equation 5
        end if
    end for
end for
```

Algorithm 3 EHO separation operator algorithm

```
for $c_i$=1, ...,CLAN do
    Substitute the worst elephant $x_{worst,c_i}$ of the clan by equation 6
end for
```

For the purpose of our work, we defined $CLAN = 1$, $\alpha = 0.9$, $\beta = 0.3$, $n_{c1} = 10$, 20, 50 and 100, and $nMaxIt = 100$ and 1000.

2.3 Genetic Algorithm

Genetic Algorithm (GA) consists of an optimization technique, inspired by the Darwinian principle of natural selection and genetic reproduction, introduced by John Henry Holland [8]. According to the Darwinian theory, the principle of selection privileges the most apt individuals, therefore, with greater probability of reproduction. The genetic codes constitute the identity of each individual, represented in the chromosomes. These principles are used in genetic algorithms, which seek the best solution for a given problem through the evolution of populations of solutions encoded by artificial chromosomes.

The artificial chromosome is a data structure that represents the individual with one of the possible solutions of the problem search space. These chromosomes are then subjected to an evolutionary process that involves assessing each individual's fitness, selection, crossover, and mutation. After several generations of evolution the population should contain the same number of individuals, but with the presence of the fittest.

A genetic algorithm is implemented through the procedure described by algorithm 4, wherein parameters *ps*, *ef* and *gn* are the population size, the expected fitness of the solution and the maximum number of generations respectively.

Algorithm 4 GA - Genetic algorithm basic cycle

```
$gen := 0$;
$population := initialpopulation()$;
$fitness := evaluate(population)$;
while ($fitness[i] \neq ef$, $1 \leq$ i $\leq ps$) & ($gen < gn$) do
    $parents := select(population)$;
    $population := mutate(crossover(parents))$;
    $fitness := evaluate(population)$;
    $gen := gen + 1$;
end while
return $fittestindividual(population)$
```

In algorithm 4, function *intialpopulation* returns a valid random set of individuals that compose the population of the first generation, while function *evaluate* returns the fitness of a given population storing the result into fitness. Function *select* chooses according to some random criterion that privilege fitter individuals, the individuals that should be used to generate the population of the next generation and functions *crossover* and *mutate* implement the crossover and mutation process, respectively, to actually yield the new population.

For implementation effect, we defined the crossover rate as 0.8, the mutation rate as 0.3, $gn =$ 100 and 1000, $ps =$ 10, 20, 50 and 100. The roulette method was used for individual selection and the representation of the cromossomes was defined in real.

3 Evaluated Features

The method proposed in this work is reached using real data of 48 transformers. These data are available in a public database published in journals and annals of conference[9], [10] and [11]. For the characterization of the transformer, six tests of quality analysis of insulating oil, introduced below, were used: water content, acidity, breakdown voltage, dissipation factor, dissolved gas analysis and 2-furfuraldehyde analysis.

3.1 Water Content

During the manufacturing process, transformers have, as standard, the variation of the concentration of moisture in the insulation between 0.5% and 1%, where the increase of the concentration of moisture is due to the degradation of the insulation or communication with the atmosphere [12]. The IEC 60814 and ASTM D1533 standards with the Karl Fisher Titration Method determine the methodologies for assessing the moisture content, commonly given in parts per million (ppm) or μl/l[13]. The determination of this test has high importance, since it is one of the main factors accelerating the aging process, reducing the dielectric capacity of the solid and liquid insulation.

3.2 Acidity

The production of acids in the isolation of power transformers results from the oxidation process, where hydrocarbons, present in the oil, react with the dissolved air, generating carboxylic acids [14]. The concentration of acids in the transformer is measured by chemical titration standardized by IEC 62021-1, IEC 62021-2 and ASTM D1534, given in mg of KOH/g [15]. The increase in acid concentration in the insulating oil occurs with extended periods in overload, being an indication of the deterioration process of the solid insulation [16][17].

3.3 Breakdown Voltage

The breakdown voltage is a measure to verify the dielectric supportability of the insulating oil, presenting significantly high values when the oil is without the presence of contaminants, such as water and solid particles [12]. The tests are carried out according to IEC 60156 and ASTM D1816, checking breakdown voltage of the oil sample in environments with a controlled temperature of 20° to 30° C, with a gap between the electrodes of 1mm [12]. The objective of the test is to verify the presence of low dielectric withstand in oil, which may lead to the increase of partial discharges, accelerating the process of aging of the transformer [9].

3.4 Dissipation Factor (tanδ)

The verification of the dissipation factor aims at quantifying the level of energy lost in the transformer during its operation [9]. The test is performed according to IEC 60247, being important to verify the energy dissipation of the insulating oil, since it is in the form of heat dissipation, which accelerates the aging process of the asset.

3.5 Dissolved Gas Analysis (DGA)

The determination of the concentration of combustible gases is one of the most important tests, responsible for indicating the evolution of gases, which, in the normal operation process of a transformer, already exist. However, with the increase of their concentration, it becomes an indication of failure or aging of the asset [4]. The evaluations can be performed through traditional criteria, such as Rogers, Dornenburg, Duval Triangle, CEGB Criteria, Laborelec Method, among others. These methods characterize types of faults within transformers by the concentration of key gases. However, we used in this work the total concentration of gases, such as H_2, CH_4, C_2H_4, C_2H_2 e C_2H_6, to determine the operational condition [9].

3.6 2-Furfuraldehyde Analysis

The analysis of concentration 2-furfuraldehyde realizes condition evaluation of solid insulation of the asset [18] [19] [20] [21]. This technique is responsible for measuring the concentration of aldehyde chains in the insulating oil. In turn, the breakdown of aldehyde chains are the result of the aging of the insulation paper. The increase of the aldehyde concentration in the insulating oil indicates the loss of mechanical properties of the paper, generating a situation favorable to the increase of partial discharges and increase of the aging process.

4 Proposed Method

Based on the tests performed on each transformer and the diagnostics, a model was established, which was implemented in four stages: preprocessing of data, application of computational intelligence techniques, validation of the solution found and application of the methodology in graphical user interface.

The input data were separated into two groups: data for the Health Index weights optimization process and data for validation. The first step in the development was the preprocessing of the data, in which normalization of the characteristics was performed based on the standard deviation of the results of each test.

After preprocessing, computational intelligence techniques were applied, *i.e.* PSO, EHO and GA, seeking the set of optimal weights (w_i) used in equation 7 for the composition of the Health Index:

$$HI = \frac{\sum_{i=1}^{n} S_i w_i}{\sum_{i=1}^{n} w_i}. \tag{7}$$

Each individual in the search space presents a vector of weights for the composition of the Health Index. These solutions were applied in a set of known assets, each solution presenting an fitness value according to the approximation with the expert result. With the optimum global solution achieved, the validation process was performed with transformers not used in the optimization process.

In the process of constructing the Health Index, the assets were classified into three critical classes: good, medium and bad. Post-composition values of the health index were, then, established, where bad assets are equipments with an index ranging from 0.75 to 1, medium assets are equipments with an index between 0.45 and 0.75, and good assets are equipments with an index between 0 and 0.45.

The solution search process was performed with a maximum number of 100 iterations in the first case (C1) and 1000 iterations in the second case (C2). The increase in the number of iterations aimed at verifying the behavior of the methods, as well as the possibility of obtaining better results with more time for the search process.

The respective experiments were repeated for each algorithm 500 times, varying the number of individuals (n) in 10, 20, 50 and 100. In this process, a variation of the amount of data was performed, checking wether one good solution was achieved in the search process with a smaller data set. Solutions were obtained, creating data relations for optimization-validation of 26-22 (54.2% - 45.8%) and 30-18 (62.5% - 37.5%) transformers. In the end, the ideal weights were found and the solutions validated. These weights were inserted into an executable processing routine with a graphical user interface, simplifying its use.

5 Results

After the proposal of the initial method developed, a comparison was made between the techniques in order to have an overview of which of the implemented ones was more efficient in terms of the precision, speed of convergence and the amount of data needed to converge to an appropriate solution. The comparison with models based on neural networks and fuzzy inference system only occurred for the accuracy of the systems presented in scientific publications in the area [22] [9].

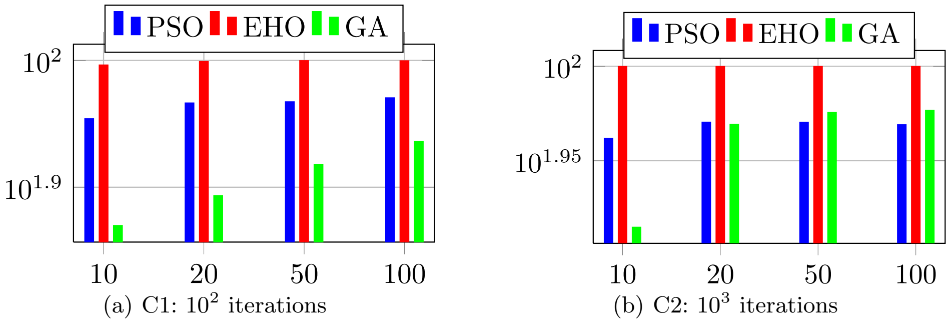
The first sequence of tests was performed with 54.2% of the transformers for training and 45.8% for validation. The values reached, in both accuracy and processing time, were average, from 500 repetitions for each configuration of the algorithms between a number of individuals and a maximum number of iterations, seen in tables 1 and 2. Figures 1 and 2 compare the performances of the techniques, using logarithmic scale.

Table 1. Accuracy with 54.2% of the transformers for training and 45.8% for validation

n	PSO		EHO		GA	
	C1 (%)	C2 (%)	C1 (%)	C2 (%)	C1 (%)	C2 (%)
10	89.97	91.61	99.19	100	74.11	82.21
20	92.58	93.44	99.82	100	78.23	93.19
50	92.78	93.43	99.98	100	82.81	94.55
100	93.45	93.15	99.95	100	86.31	94.79

Table 2. Processing time with 54.2% of the transformers for training and 45.8% for validation

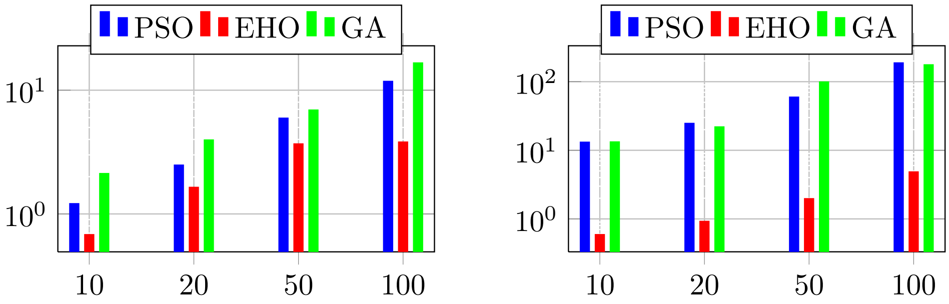
n	PSO		EHO		GA	
	C1 (s)	C2 (s)	C1 (s)	C2 (s)	C1 (s)	C2 (s)
10	1.21	13.11	0.68	0.59	2.12	13.24
20	2.48	24.65	1.64	0.92	3.96	21.92
50	5.94	59.50	3.68	1.97	6.91	99
100	11.78	187.69	3.81	4.84	16.60	176.1



(a) C1: 10^2 iterations (b) C2: 10^3 iterations

Fig. 1. Performance of the techniques for accuracy related to table 1

In order to evaluate the impact of the data in the weight optimization process, the number of training data was increased to 62.5%, reducing the number of validating data to 37.5%. The results, in relation to the accuracy and processing time, are shown in the tables 3 and 4. Figures 3 and 4 compare the performances of the techniques, using logarithmic scale.



(a) C1: 10^2 iterations

(b) C2: 10^3 iterations

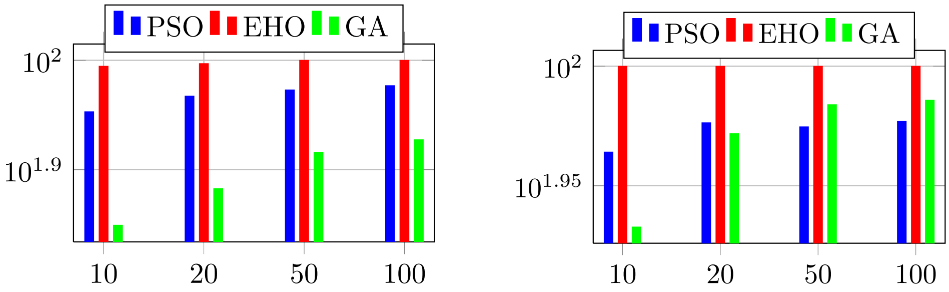
Fig. 2. Performance of the techniques for processing time related to table 2

Table 3. Accuracy with 62.5% of the transformers for training and 37.5% for validation

n	PSO		EHO		GA	
	C1 (%)	C2 (%)	C1 (%)	C2 (%)	C1 (%)	C2 (%)
10	89.72	92.08	98.74	100	70.67	85.67
20	92.74	94.71	99.28	100	76.32	93.72
50	93.94	94.34	99.98	100	82.37	96.37
100	94.78	94.84	99.97	100	84.59	96.80

Table 4. Processing time with 62.5% of the transformers for training and 37.5% for validation

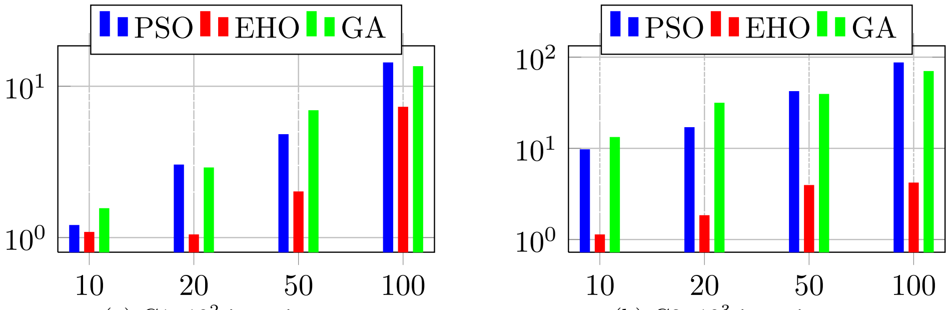
n	PSO		EHO		GA	
	C1 (s)	C2 (s)	C1 (s)	C2 (s)	C1 (s)	C2 (s)
10	1.20	9.61	1.08	1.12	1.55	13.11
20	3.01	16.81	1.04	1.82	2.88	31.15
50	4.78	41.78	2.00	3.90	6.88	38.93
100	14.24	86.21	7.25	4.15	13.45	69.38



(a) C1: 10^2 iterations

(b) C2: 10^3 iterations

Fig. 3. Performance of the techniques for accuracy related to table 3



(a) C1: 10^2 iterations

(b) C2: 10^3 iterations

Fig. 4. Performance of the techniques for processing time related to table 4

According to the literature, Health Index models, with the same characteristics presented in the present work, reached 96.55% of accuracy when using artificial neural networks [22] and 96.7% of accuracy when using fuzzy inference systems [9].

6 Conclusions

The proposed method aimed at the classification and prioritization of high power transformers, performing the composition of the Health Index with computational intelligence techniques, such as PSO, EHO and GAs. The results obtained, with relation to accuracy and processing time, proved the ability of these techniques in providing very good results, with a high degree of precision, reproducibility of results and without the presence of randoness in the definition of the weights of the Health Index.

It is worth mentioning that the convergence time for results with computational intelligence is low, compared to other optimization methods. It was observed that the implementation with the EHO algorithm proved to be very efficient and adequate to solve the problem, surpassing the implementations with PSO and GA, with average precision equal to 100% and average processing time of less than 1s in the best solution found.

Regarding the data relationship for solution optimization and validation, even with few data to search for weights, algorithms based on computational intelligence presented good solutions, being able to generalize and characterize patterns in an appropriate way. With the increase of the number of individuals in the algorithms, there was a slight improvement in the results, presenting good solutions even with few individuals and, thus, reducing computational cost and processing time.

Fuzzy inference systems and neural networks were compared only in terms of accuracy, since the literature does not meet the minimum reproduction requirements of the methodology. However, the method proposed in this work surpasses the two models in terms of flexibility, being easily expandable, consuming less time for data processing due to the characteristics of algorithms based on computational intelligence.

As for future work, we intend to investigate other computational intelligence techniques, such as Artificial Bee Cololy, Cuckoo Search, Bacterial Foraging Optimization, Firefly, Fireworks Algorithm for Optimization.

References

1. Jahromi, A., Piercy, R., Cress, S., Service, J., Fan, W.: An approach to power transformer asset management using health index. IEEE Electrical Insulation Magazine **25** (2009) 20–34
2. ONS: Sin performance indicators. http://ons.org.br/. Accessed 28 November 2017 (2017)
3. Wang, M., Vandermaar, A., Srivastava, K.D.: Review of condition assessment of power transformers in service. IEEE Electrical Insulation Magazine **18** (2002) 12–25
4. Abu-Elanien, A.E., Salama, M.: Asset management techniques for transformers. Electric power systems research **80** (2010) 456–464

5. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Neural Networks, 1995. Proceedings., IEEE International Conference on. Volume 4. (1995) 1942–1948 vol.4
6. Engelbrecht, A.P.: Fundamentals of Computational Swarm Intelligence. John Wiley & Sons, New Jersey (2005)
7. Wang, G.G., Deb, S., d. S. Coelho, L.: Elephant herding optimization. In: 2015 3rd International Symposium on Computational and Business Intelligence (ISCBI). (2015) 1–5
8. Holland, J.: Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence. University of Michigan Press (1975)
9. Abu-Elanien, A.E., Salama, M., Ibrahim, M.: Calculation of a health index for oil-immersed transformers rated under 69 kv using fuzzy logic. IEEE Transactions on Power Delivery **27** (2012) 2029–2036
10. Cerón, A.F., Echeverry, D.F., Aponte, G., Romero, A.A.: Índice de salud para transformadores de potencia inmersos en aceite mineral con voltajes entre 69kv y 230kv usando lógica difusa. Información tecnológica **26** (2015) 107–116
11. Chacón-Troya, D.P., Lata, J.P., Medina, R.D.: Health index assessment for power transformers with thermal upgraded paper up to 230kv, using fuzzy inference. part ii: A sensibility analysis. In: Devices, Circuits and Systems (ICCDCS), 2017 International Caribbean Conference on, IEEE (2017) 109–112
12. Standard, I., et al.: Mineral insulating oils in electrical equipment–supervision and maintenance guidance. BS EN **60422** (2013)
13. Lewand, L.: Understanding water in transformer systems. Neta World (2002) 1–4
14. Erdman, H.G.: Electrical insulating oils. Volume 998. ASTM International (1988)
15. International, A.: Standard Test Method for Acid and Base Number by Color-indicator Titration. ASTM International (2012)
16. Wahab, M.A., Hamada, M., Zeitoun, A., Ismail, G.: Novel modeling for the prediction of aged transformer oil characteristics. Electric Power Systems Research **51** (1999) 61–70
17. Lundgaard, L.E., Hansen, W., Linhjell, D., Painter, T.J.: Aging of oil-impregnated paper in power transformers. IEEE Transactions on power delivery **19** (2004) 230–239
18. Kachler, A.J., Hohlein, I.: Aging of cellulose at transformer service temperatures. part 1: Influence of type of oil and air on the degree of polymerization of pressboard, dissolved gases, and furanic compounds in oil. IEEE Electrical Insulation Magazine **21** (2005) 15–21
19. Pradhan, M., Ramu, T.: Criteria for estimation of end of life of power and station transformers in service. In: Electrical Insulation and Dielectric Phenomena, 2004. CEIDP'04. 2004 Annual Report Conference on, IEEE (2004) 220–223
20. Van Bolhuis, J., Gulski, E., Smit, J.: Monitoring and diagnostic of transformer solid insulation. IEEE Transactions on Power Delivery **17** (2002) 528–536
21. Emsley, A., Xiao, X., Heywood, R., Ali, M.: Degradation of cellulosic insulation in power transformers. part 3: Effects of oxygen and water on ageing in oil. IEE Proceedings-Science, Measurement and Technology **147** (2000) 115–119
22. Abu-Elanien, A.E., Salama, M., Ibrahim, M.: Determination of transformer health condition using artificial neural networks. In: Innovations in Intelligent Systems and Applications (INISTA), 2011 International Symposium on, IEEE (2011) 1–5

Application of the surrogate models for protein structure prediction

Hojjat Rakhshani, Lhassane Idoumghar, Julien Lepagnot, Mathieu Brévilliers

Université de Haute-Alsace, IRIMAS-UHA, F-68093 Mulhouse, France
{hojjat.rakhshani, lhassane.idoumghar, julien.lepagnot, mathieu.brevilliers}@uha.fr

1 Introduction

The protein structure prediction (PSP) is defined as finding 3D dimensional structure of a protein from its primary sequence. The meta-heuristic algorithms have shown promising results to cope with this problem [1]. The main challenge then came from computationally expensive simulations. The surrogate models are offered as effective tools to address such computationally expensive objective tasks. Accordingly, this study introduces a preliminary reference on the importance of surrogate-models for the PSP. Toward this goal, we introduce a model management framework which combines Stochastic Response Surface (SRS) [2] and differential evolution (DE) [3] to interleave the exact and approximate objective functions during the search process. The proposed approach is then applied on several variants of DE.

2 AB off-lattice protein structure prediction

The main step in PSP involves explaining relation of the tertiary structure in a protein to its free energy. One of the best and realistic models is AB Off-Lattice [4] which takes into the account sequence independent and dependent local interactions. In this model, amino acids are classified either as hydrophobic (A) or polar (B) residues. These monomers are linked to each other by rigid unit-length bonds to form the protein structure. In this way, the structure of a protein can be determined by the bond angles $[\theta_2, \theta_3, \cdots, \theta_{n-1}]$, where n demonstrates the number of bonds. The free energy function of a sequence of amino acids is then given as below [4]:

$$\mathrm{E} = \sum_{i=2}^{n-1} \frac{1-\cos(\theta_i)}{4} + 4\sum_{i=1}^{n-2}\sum_{j=i+2}^{n} \left[r_{i,j}^{-12} - \varsigma(\xi_i,\xi_j) r_{i,j}^{-6}\right] \tag{1}$$

where $r_{i,j}$ is the distance between monomer i and j in the chain as given in (2).

$$r_{i,j} = \sqrt{\left[1 + \sum_{k=i+1}^{j-1} \cos\left(\sum_{l=i+1}^{k} \theta_l\right)\right]^2 + \left[\sum_{k=i+1}^{j-1} \sin\left(\sum_{l=i+1}^{k} \theta_l\right)\right]^2} \tag{2}$$

Also, $\varsigma(\xi_i, \xi_j)$ denotes interaction between the two monomers i and j:

$$\varsigma(\xi_i,\xi_j) = \frac{1}{8}(1 + \xi_i + \xi_j + 5\xi_i\xi_j) \tag{3}$$

In such schema, ξ_i and ξ_j are two discrete variables that denote residue species. Accordingly, coefficient $\varsigma(\xi_i, \xi_j)$ is equals to 1 for AA pairs, 0.5 for BB pairs, and -0.5 otherwise. More details about the described model can be found in [4].

3 The proposed approach

The proposed approach, we called it as DE-SRS, begins with an population initialization step which address poor uniformity of the initial individuals in the DE algorithm. More precisely, DE-SRS adopts Latin Hyper-cube Sampling (LHS) [5] which is a statistical method. This sampling scheme can enhance the chance of exploring promising regions of the search space on high dimensional problems with a minimum number of samples. All the generated solutions $\mathbf{x}_i : i \in [1, n]$ are then evaluated using the exact fitness function f. The data points $\mathcal{B}_n \leftarrow \{(\mathbf{x}_i, f(\mathbf{x}_i)) : i \leftarrow 1 \cdots n\}$ are used to initialize the population in DE and also to train our surrogate model. Among different surrogate models, we utilized the Radial Basis Function (RBF) which is a good model for high dimensional problems [6]. Next, DE algorithm evolves the solutions only for a single iteration. Similar to the initialization step, the exact objective function f will be used to evaluate the solutions. Subsequently, the DE-SRS continues with SRS local optimization method to incorporate the approximation values of the exact functions for evolving the solutions. The SRS tries to obtain an optimal response for a surface (output variable) which is influenced by several explanatory variables (input variables). To do so, it uses a collection of mathematical and statistical information from surrogates and exact fitness function as described in [2]. All the solutions obtained by SRS are compared to the solutions found by the DE algorithms. If the local SRS method performs better than the DE, the former replaces the latter solutions. Otherwise, the solutions obtained by the SRS are discarded.

Appropriate switching between the local and global search techniques is an important issue for the DE-SRS. It is of interest to stop the global optimizer earlier in order to reduce the computational time. However, this strategy increases the probability of falling into a local minimum [7]. To cope with this issue, Duan et al [7] conducted a study in order to identify the most efficient way of applying a local search method embedded in a global optimization algorithm. They found that applying a local search method to every generated solution with a distinct probability is the most efficient method. So, we adopted the aforementioned approach in the DE-SRS. According to this approach, the solutions obtained by the stochastic explorer will undergo the deterministic search with a certain probability Pa.

After evolving the individuals using the DE-SRS, we train our RBF surrogate model using these obtained individuals. The DE-SRS switches between DE, SRS and surrogate training phases until stopping criteria are met.

4 Experimental results

In this section, performance of the DE-SRS on two protein sequences, namely 1AGT and 1AHO, from protein data bank [8] is investigated. In DE, F is 0.5, population size is 100 and crossover rate CR is 0.9. The introduced DE-SRS algorithm applied on DE with different mutation strategies: DE/best/1, DE/rand/1, DE/rand-to-best/1, DE/best/2 and DE/rand/2. From our simulations, we also found that $Pa = 0.2$ is an appropriate configuration. All simulations are performed using Matlab. For the SRS, we adopted the same implementation and parameter configuration as suggested in [2].

The experiments are repeated 10 times for each problem. The number of function evaluations and runs are determined as 100000 and 10, respectively. It is also worth mentioning that during each local search phase, the SRS uses 50 exact function evaluations. A non-parametric test, called Kruskal-Wallis, followed by the Tukey-Kramer multiple comparison procedure is also conducted to determine whether the obtained results by DE-SRS are significantly different from the standard algorithm. To do so, the average values obtained from 10 runs are subjected to this test with $alpha = 0.05$ as the level of significance. In this study, we use symbol ”+” to indicate that with 95% certainty DE-SRS presented a superior performance.

The obtained results are presented in Table 1 and Table 2. The comparison is performed based on different extensions of the DE. From these tables, we can see that the DE-SRS yields superior results for both proteins, independent of the introduced mutation strategy. The obtained results confirm our hypothesis from Section 1, that the surrogate models could be effective for the PSP problem.

5 Conclusion

This study investigated the application of surrogate models for the PSP problem by combining the DE and SRS algorithms. To provide a perspective for present knowledge and future research, we considered different variants of the DE. Experimental studies are carried out on two real-world PSP problems. The Kruskal-Wallis statistical test followed by the Tukey-Kramer multiple comparison procedure is also conducted. From the results derived by this research, the surrogate modeling is shown to be capable of speeding up the search process for the PSP problems.

Table 1. The best energies found for real protein sequence 1AGT

Algorithms	Mutation	Mean	Median	Best	Worst	Std.	Sign
DE	DE/best/1	1.582e+01	4.003e+00	9.883e-01	6.719e+01	2.875e+01	+
DE-SRS		-2.716e+00	-3.491e+00	-5.250e+00	3.481e+00	3.554e+00	
DE	DE/rand/1	1.777e+07	1.448e+07	1.576e+06	3.943e+07	1.510e+07	+
DE-SRS		2.178e+00	1.217e+00	-9.678e-02	6.918e+00	2.779e+00	
DE	DE/rand-to-best/1	7.597e+00	3.539e+00	-1.909e+00	3.464e+01	1.355e+01	+
DE-SRS		-4.591e+00	-4.059e+00	-6.354e+00	-2.909e+00	1.404e+00	
DE	DE/best/2	6.193e+07	3.876e+07	1.924e+07	1.732e+08	6.294e+07	+
DE-SRS		1.009e+02	2.998e+01	4.980e+00	3.939e+02	1.656e+02	
DE	DE/rand/2	7.970e+08	8.310e+07	1.753e+07	3.378e+09	1.455e+09	+
DE-SRS		4.647e+04	2.123e+04	2.511e+02	1.785e+05	7.492e+04	

Table 2. The best energies found for real protein sequence 1AHO

Algorithms	Mutation	Mean	Median	Best	Worst	Std.	Sign
DE	DE/best/1	7.872e+04	7.107e+04	3.996e+01	6.719e+01	6.964e+04	+
DE-SRS		-4.145e+00	-4.500e+00	-4.893e+00	-2.969e+00	7.696e-01	
DE	DE/rand/1	4.360e+10	2.783e+10	2.942e+08	1.975e+11	5.904e+10	+
DE-SRS		9.182e+00	4.885e-01	-5.092e-01	8.296e+01	2.596e+01	
DE	DE/rand-to-best/1	4.373e+05	1.072e+05	2.835e+02	2.308e+06	7.105e+05	+
DE-SRS		-5.089e+00	-5.242e+00	-6.755e+00	-4.047e+00	7.633e-01	
DE	DE/best/2	8.395e+09	1.134e+10	1.869e+09	1.197e+10	5.660e+09	+
DE-SRS		2.275e+02	1.091e+01	2.433e+00	6.693e+02	3.826e+02	
DE	DE/rand/2	7.872e+04	7.107e+04	3.996e+01	1.806e+05	6.965e+04	+
DE-SRS		6.146e+00	6.520e+00	5.193e+00	7.969e+00	7.696e-01	

References

1. B. Bokovi and J. Brest, "Differential evolution for protein folding optimization based on a three-dimensional AB off-lattice model," Journal of molecular modeling, vol. 22, p. 252, 2016.
2. R. G. Regis and C. A. Shoemaker, "A stochastic radial basis function method for the global optimization of expensive functions," INFORMS Journal on Computing, vol. 19, pp. 497-509, 2007.
3. R. Storn and K. Price, "Differential evolutiona simple and efficient heuristic for global optimization over continuous spaces," Journal of global optimization, vol. 11, pp. 341-359, 1997.
4. F. H. Stillinger and T. Head-Gordon, "Collective aspects of protein folding illustrated by a toy model," Physical review E, vol. 52, p. 2872, 1995.
5. M. D. McKay, R. J. Beckman, and W. J. Conover, "Comparison of three methods for selecting values of input variables in the analysis of output from a computer code," Technometrics, vol. 21, pp. 239-245, 1979.
6. A. Daz-Manrquez, G. Toscano-Pulido, and W. Gmez-Flores, "On the selection of surrogate models in evolutionary optimization algorithms," in Evolutionary Computation (CEC), 2011 IEEE Congress on, pp. 2155-2162, 2011.
7. Q. Duan, T. W. Liao, and H. Yi, "A comparative study of different local search application strategies in hybrid metaheuristics," Applied Soft Computing, vol. 13, pp. 1464-1477, 2013.
8. H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, et al., "The protein data bank, 1999," in International Tables for Crystallography Volume F: Crystallography of biological macromolecules, Springer, pp. 675-684, 2006.

A Hybrid Genetic Algorithm for the job shop problem with transportation and blocking no wait constraints

S. Louaqad[1], O. Kamach[2] and A. Iguider[3]

1. LTI, ENSAT, Abdelmalek Essadi University, Tanger, Maroc
louqad@gmail.com

2. LTI, ENSAT, Abdelmalek Essadi University, Tanger, Maroc
Kamach@ensat.ac.ma

Keywords: job shop, Transport, Blocking, No wait, combinatorial optimisation, genetic algorithm.

Abstract

The problem studied in this paper is the Transportation job shop problem with blocking and no wait constraints. This problem is an extension of the classical job shop problem that take into account transport operations and the absence of storage space between machines. We formulate the problem by means of a new disjunctive graph. The new disjunctive graph served us to develop a Hybrid Method based on Genetic algorithm and greedy heuristics. The genetic Algorithm is used to evolve operations sequencing. The greedy Heuristic is then used to assign robot to transport operations. Computational results for a set of benchmarking tests are reported and the effectiveness of our methods are discussed.

1 Introduction

The classical job-shop problem is known as a standard problem in scheduling and has been widely investigated over the last few decades [15–16]. However, in real world, several problems often cannot be modeled as a classical job-shop problem, due to additional features. This is especially the case in flexible manufacturing systems where the model set by researchers have to take into account some aspects such as material handling, storage space etc.

The jobs shop problems with transportation, blocking and no wait constraints (NWBT JSSP) are met for example in factories with robotic cells and no buffers between machines. A robotic cell is a flow-shop or job-shop scheduling cell in which the jobs are transported from machine to machine by one or more robots. We have to assign the transport operations to the robots and to schedule both the machine and robot operations. In some kind of robotic cells, there is no buffer between machines and jobs must be conducted from one machine to another one with no interruption, so we have to deal with blocking no wait constraints. Robotic cells with no storage buffers between machines are widely used in practice[12], especially in steelmaking and chemical batch production [13] [14].

Several researchers studied the Blocking Job shop Scheduling Problem BJSSP and the No Wait Job Shop Scheduling Problem (NWJSSP)[2]. [3] describes several applications of machine scheduling with blocking and no wait in process and reviews the computational complexity of a variety of related problems. [4] and [5] formulate these problems by means of alternative graphs. [7] develop a genetic algorithm for solving no-wait and Blocking Job Shop problems (NWB JSSP) and [8] and [9] introduce a local search approach for the generalized Blocking Job Shop problem with application in automated warehouses. [6] study a multi-resource job shop problem with blocking constraints. [10] propose a tabu search algorithm to solve the BJSSP for cyclical scheduling. [11] proposes a combination of a branch and bound algorithm with alternative graphs and develops two methods based on genetic algorithms to solve the BJSSP. [38,39] present three improved heuristics basing on two simple construc-tive heuristics and a mimetic algorithm, respectively. [40] proposed a multi-search parallelization approach based on master/worker paradigm, exploiting the multi-Core CPU-processors to solve optimally a BJSSP.

Several researchers have devoted to study job shop scheduling problems with transportation constraints (TJSSP) in various systems. However, the progress is limited as this kind of problem is difficult to solve even for simplified and small size cases. A State of-the-Art Review and Classification Schema for the Job Shop Scheduling Problem with Transportation Resources is given by [34]. [19] developed a mixed integer programming (MIP)formulation raising this constraint on the vehicles. [29] used a mixed integer linear program (MILP) to find optimal solutions for the Flexible Manufacturing Systems Scheduling Problem with one vehicle. [31] studied coupled task problem and one-machine robotic cell problems. It reported new algorithmic procedure for this problem with or without tolerances on the distance. [32] applied a decomposition method where the master problem (scheduling) is modeled with constraint programming and the subproblem (conflict free routing) with mixed integer programming. [30] proposed a polynomial algorithm for finding the optimal cycle in a robotic cell with production of identical parts. [17] integrated transport constraints in the scheduling problem with one robot. [28] considered a cyclic hoist scheduling problem with a single hoist, but without assignment problem. [18] proposed a dynamic programming approach to construct optimal machine and vehicle schedules. [20] and [21] elaborated a genetic algorithm. [22] and [23] proposed, respectively, neural networks and tabu search approaches. [24] described a hybrid method composed of a genetic algorithm for the scheduling of machines and a heuristic for the scheduling of vehicles. [25] and [26] considered a job shop problem with several robots, with fixed operation times andfixed assignment of machine for each job's operation. [27] studied a two machines flow shop scheduling problem with intermediate transportation with a single transporter. [41] proposed a hybrid metaheuristic approach based on clustered holonic multiagent model for the JSPT-MR. [42] considered a coloured Petri net-based hybrid heuristic search approach to simultaneous scheduling of machines and automated guided vehicles

To the best of our Knowledge there is no research that incorporate simultaneously transport, blocking and no wait constraints in the problem of job shop scheduling. Two common approaches to tackle the Job Shop scheduling problems with additional constraints are the utilization of exact methods and heuristic approaches [1] and [33]. We have already used a MILP to solve The NWB JSSP Problem [47]. The MILP allow us to find optimal solutions but is not suitable for large size instances. In this paper, we propose a hybrid method of genetic algorithm and a greedy heuristic based on priority rules. The genetic Algorithm is used to evolve operations sequencing. The greedy Heuristic is then used to assign robot to transport operations. To evaluate solutions, we elaborate a modified disjunctive graph which contains transportation nodes, alternative and perishability arcs.

This paper is structured as follows. In the next section, we will define our problem and the notations associated to its formulation. After, in Section 3, we present the disjunctive graph representation for the NWBT JSSP. In section 4, we describe and present our Hybrid Genetic Algorithm. Section 5 discusses a series of experimental tests and computational results. Conclusion follows in section 6.

2 PROBLEM DEFINTION

We consider a job shop problem with several transport robots and no buffers. In this problem, a set of n jobs $\{J_1, J_2, \ldots, J_n\}$ are processed on a set of m machines $\{M_1, M_2, \ldots, M_m\}$ and transported by a set of k $\{r_1, r_2, \ldots, r_k\}$. Transportation times are robot-independent. Every job J_i require an operation order, $\{J_i=\{O_{i1}, O_{i2} \ldots, O_{in_i}\}\}$, that must be executed according to its manufacture process. Operation O_{ij} of the job J_i requires the exclusive use of $M_l(1 \in \{1,2,\ldots m\})$ for an uninterrupted duration p_{ij}, its processing time; the preemption is not allowed; each machine can process only one job at a time; the machine which execute the operation O_{ij} is denoted as M_{ij}. In addition, we consider transportation operations between two machines. Consider two successive operations of the same job O_{ij} and O_{ij+1} to execute in two machines M_{ij} and M_{ij+1}. T_{ij} is used to denote transport operation of job J_i from machine M_{ij} to machine M_{ij+1}. Each robot can handle at most one job at one time. Loaded transfer times do not depend on the job transported, but only on the travel routes and the robot which perform the transportation operation. These times are given by C^{r}_{pl}

where r represents the robot and p,l represents the route between machine M_p and M_l. It is assumed that the triangle inequality is satisfied:

$\forall \{p, l, h\} \in \{1,2,\ldots,m\}$ machine indexes.

$\forall\, r \in \{r_1, r_2, \ldots, r_k\}$

$$C^{r}_{ph} + C^{r}_{hl} \geq C^{r}_{pl} \quad (1)$$

(1) means that the direct way between two machines is at least as short as the detour through a third machine. Otherwise, the robot always takes the shorter way through the third machine.

Note that a sequence of loaded transport operations indirectly induces necessary empty moves. Empty transfer time from machine M_p to M_l is denoted V^r_{pl}. It is assumed that:

$$\forall \{p, l\} \in \{1,2,\ldots,m\} \text{ machine indexes}$$

$\forall \{r, r'\} \in \{r_1, r_2, \ldots, r_k\}$

$$\begin{cases} V^r_{pp} = 0 \\ V^r_{ph} + V^r_{hl} \geq V^r_{pl} \\ C^{r'}_{pl} \geq V^r_{pl} \end{cases} \qquad (2)$$

The first assumption means that no empty transfer time is considered if a robot waits at the same machine the next transportation operation. The second one is the triangular inequality for empty moves. The third one means that empty transfers between two machines by a robot *r* take less time than loaded transfers between this two machines by another robot *r'*. (It is also valid if *r* = *r'*). In the other hand, we consider the blocking constraint because there is no machine buffer. This means that after finishing its processing on the machine, a job has to stay there until it is unloaded by the robot. During this stay, the machine is blocked and not available for processing any other job. We also consider the no wait constraint that means if the robot transporting the job J_i reaches machine M_{ij}, the operation O_{ij} must start immediately without any interruption.

We distinguish two cases of blocking: blocking with swap allowed and blocking no swap [37]. In our case, we consider that the job can move independently and therefore the swaps are allowed. It means that whenever there is a job J_i in Machine M_l and a Job $J_{i'}$ in a Robot r_s , each one waiting for another to be freed as the job J_i has to be transferred from Machine M_l to robot r_s and job $J_{i'}$ from robot r_s to machine M_l , Job J_i and $J_{i'}$ could move simultaneously.

The scheduling problem objectives are:

- To determine the starting time d_{ij} for each machine operation O_{ij}. (the completion time is denoted f_{ij}).
- To assign a handling robot to each transport operation T_{ij} and to determine its starting time d'_{ij}. (the completion time is denoted f'_{ij})
- To minimize the Makespan denoted C_{max} = max (C_i) where C_i denotes the completion time of the last operation of job J_i .

All data p_{ij}, C^r_{pl}, V^r_{pl}, d_{ij}, d'_{ij}, f_{ij}, f'_{ij}, and C_{max} are assumed to be non-negative integers.

3 DISJUNCTIVE GRAPH MODEL

In this section, the disjunctive graph model that we will use as a basis for developing our two Heuristics is described. It is an extension of the classical disjunctive graph model *G = (N, A, E)* [35] in order to take into account transportation and the blocking no wait constraint. [17] extend the disjunctive graph model for classical job shop to correspond to $G = (V, C, D_M, D_R)$ to describe the classical job shop problem with transportation operation performed by a single robot. [25] extend it to $G = (V_M, V_T, C, D_M, D_R)$ to encompass several robots. The disjunctive graph $G = (V_M, V_T, C, D_M, D_R)$ dedicated to job shop problem with transportation by several robot consists of: a set of vertices V_M containing all machine operations; a set of vertices V_T representing the set of transport operations obtained by an assignment of one robot to each transport operation ; two dummy nodes *{0}* and *{*}*. The graph consists also of a set of conjunctive arcs *C* representing precedence constraints between operations of the same job, a set D_M of disjunctive arcs connecting the operations to be processed by the same machine and a set D_R of disjunctive arcs connecting the transport operations to be processed by the same robot. [11] introduced Alternative graph formulation to model the blocking constraints in the classical job shop.

Since the graph of [25] already deals with all conflicts regarding job-shop machines and transport operations we will use this graph to incorporate blocking and no wait constraints.

To deal with the blocking constraint in the job shop with transportation, each disjunctive arc in the set D_M is replaced by a disjunctive couple. More precisely, for each pair of operation O_{ij} and $O_{i'j'}$ sharing common machine, we introduce a disjunctive couple of two arcs: one from vertex T_{ij} to $O_{i'j'}$ and another one from $T_{i'j'}$ to O_{ij}. Since the swap is allowed, the weight of the disjunctive couples is zero.

To deal with no wait constraints, for each two consecutive transport and machine operations T_{ij} et O_{ij+1}, we add to the Set *C* a negative arc between these two operations, The weight of the negative arc is $-t_{ij}$. This negative arc assume that a machine operation must start processing immediately after the completion of transport operation.

The Figure 1 represents the non-oriented disjunctive graph of the NWBT JSSP *(P1)* of size (*3* jobs × *3* machines × *4* robots) defined by table 1:

Table 1: The NWBT JSSP (P1) instance

- Machine operations:	- On load transfer times	- empty transfer times:
J_1 [M_1:8 ; M_2:10 ; M_3:6] ; *J_2 [M_2:14 ; M_3:10 ; M_1:10] ;* *J_3 [M_1:14 ; M_3:10 ; M_2:8] ;*	*M_1 [M_1:0 ; M_2:2 ; M_3:4] ;* *M_2 [M_1:2 ; M_2:0 ; M_3:2] ;* *M_3 [M_1:4 ; M_2:2 ; M_3:0]*	*M_1 [M_1:0 ; M_2:1 ; M_3:2] ;* *M_2 [M_1:1 ; M_2:0 ; M_3:1] ;* *M_3 [M_1:2 ; M_2:1 ; M_3:0] ;*

To solve the scheduling problem it is necessary to select one arc from each couple of the Set D_M to assign one robot to each transport operation and to turn all undirected arcs between robots into directed ones. We suppose that *(S1)* defined by table 2 is the solution of the problem *(P1)*.. This solution *(S1)*is represented by the conjunctive Graph of the figure 2.

Table 2: Solution (S1) of the NWBT JSSP (P1)

Machine 1	$\{O_{11}, O_{31}, O_{23i}\}$	Robot 1	$\{T_{11}, T_{22}, T_{32i}\}$
Machine 2	$\{O_{21}, O_{12}, O_{23i}\}$	Robot 2	T_{12}
Machine 3	$\{O_{22}, O_{32}, O_{13i}\}$	Robot 3	T_{21}
		Robot 4	T_{31}

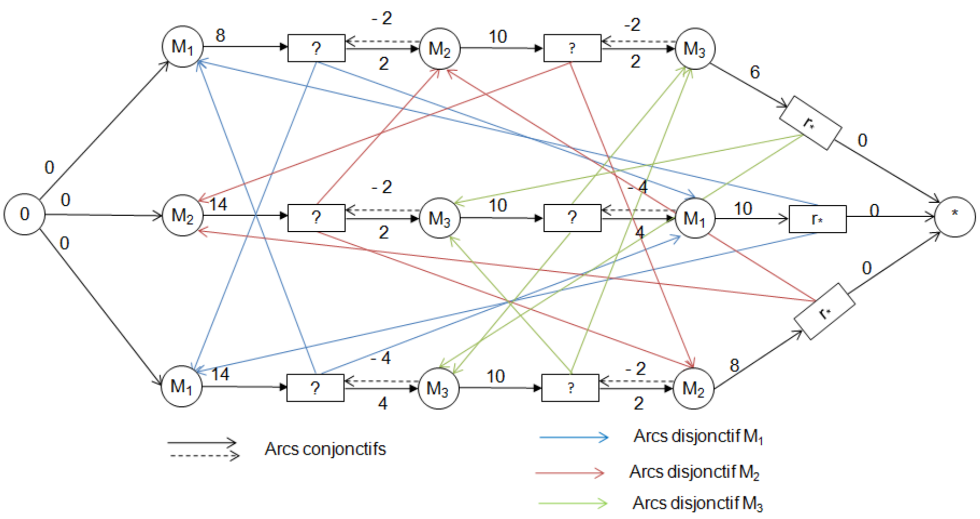


Figure 1. Disjunctive Graph representing the problem (P1)

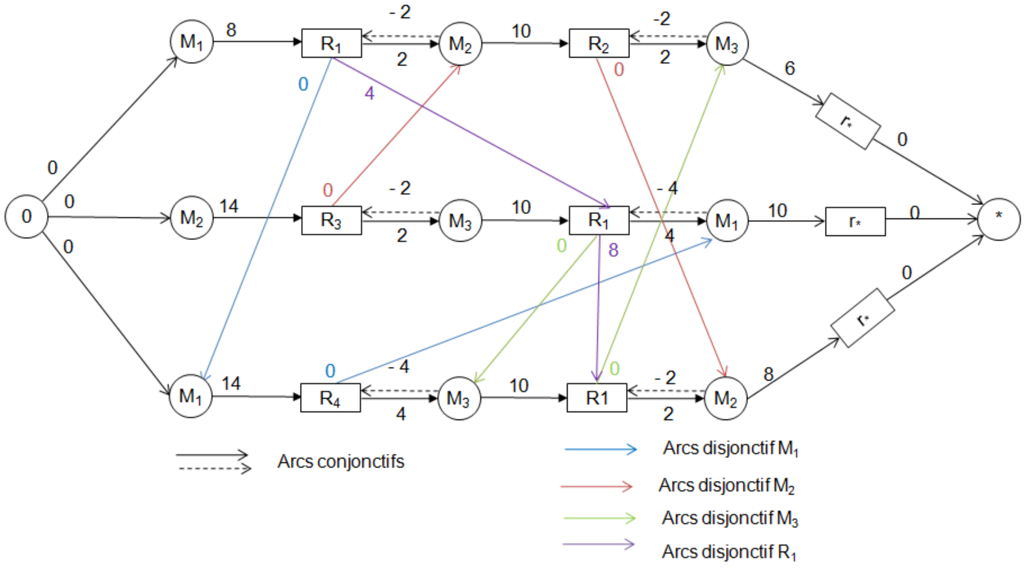


Figure 2: The conjunctive Graph representing the solution (S1) of the problem (P1)

4 The Hybrid Genetic Algorithm

4.1 Genetic Algorithm principles

Genetic algorithms (GAs) were developed during the 1960s [44]. They are stochastic search heuristics that mimic biological evolution. A genetic algorithm has a population of individuals (called chromosomes or genotype). Individuals who fit the environment best should have a better chance to propagate their offspring. By the same reason, solutions that have the best ‘‘fitness” should receive higher probability to search their ‘‘neighbors”. This idea was first proposed by [45].

The five main components of a GA are listed below[46]:

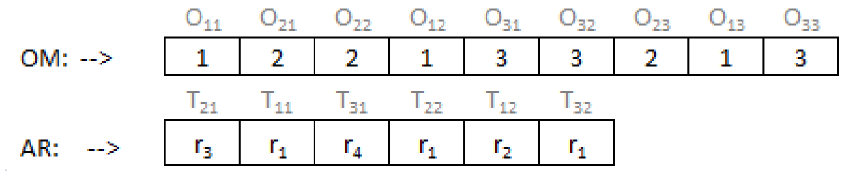
(1) Parameters setting for the algorithm, the operators and so forth.
(2) A way of encoding solutions to the problem – fixed length string of symbols.
(3) A way of initializing the population of solutions.
(4) Operations that may be applied to parents such as reproduction, crossover, mutation, and other domain specific operators.
(5) An evaluation function that returns a rating for each solution.

When problem-specific mechanism is encapsulated in component (4), either a local search procedure or a heuristic tailored for the problem, it is usually called a hybrid genetic algorithm

4.2 Chromosome representation and population initialization

Chromosome representation:

When solving problems with genetic algorithms, the first step is to represent a solution to a problem as chromosome. A good representation is crucial because it significantly affects all the steps of the algorithm and has a great impact on computational time. In our representation, the chromosome is encoded using a selection of Two Vectors (*OM* and *AR*) as shown in figure 3:



	O_{11}	O_{21}	O_{22}	O_{12}	O_{31}	O_{32}	O_{23}	O_{13}	O_{33}
OM: -->	1	2	2	1	3	3	2	1	3

	T_{21}	T_{11}	T_{31}	T_{22}	T_{12}	T_{32}
AR: -->	r_3	r_1	r_4	r_1	r_2	r_1

Figure 3: The conjunctive Graph representing the solution (S1) of the problem

- The *OM* vector represents machine operation sequencing: We use a permutation with repetition of job numbers as it has been proposed by [43] for the classical Job shop Problem. This representation has the advantage of not producing illegal sequences of each job operations.
- The vector *AR* is used to represent the robot assignments to transport operations. As we deal with The NWBT JSSP problem, the transport operation sequencing is directly deduced from vector *OM*.

Chromosome feasibility:

The permutation with repetition cannot produce illegal sequence in regards of the technological operation sequence operations of each job but, unlike classical job shop, we can obtain unfeasible schedules containing deadlock.

For an *OM* vector to represent a feasible schedule, the machine operations sequence must respect the following condition

Non deadlock condition:

Note $P(OM,O_{ij})$ the rank of O_{ij} on OM vector. Consider O_{ij}.and O_{lq} such that $M_{ij} = M_{lq}$ and $P(OM,O_{ij}) < P(OM,O_{lq})$:

OM is feasible if and only if $P(OM,O_{ij}) < P(OM,O_{ij})$

To avoid unfeasible schedules, we verify the feasibility of *OM* vector and repair its sequence so as to respect the non-deadlock condition.

Chromosome Fitness:

The solution associated with the chromosome C (a selection) is obtained by creating the graph associated with this representation. We apply the Bellman Ford algorithm to calculate the longest path distance wich correspond to the Cmax. The Fitness of C is defined as follows: Fitness (C) = 1/Cmax.

Population Initialization:
Initial population serves as the starting point of GA's searching process and it can be generated randomly or by some problem specific heuristics. we choose to generate the initial population P(0) by generating Sp random permutations of OM and AR. Sp is the size of P(0).

4.3 Greedy Heuristic for robot assignment

After setting OM vector, the scheduling of transport operations is deduced directly. To set the Vector AR representing the assignment of robots to transport operations, we propose a dedicated heuristic *(HC)* that complete iteratively AR vector according to priority rules.

These Selection rules of robot assignment are as follows:

- Rule 1: To select the robot that will perform the transportation operation T_{ij}, we opted to choose the robot which provides the minimal completion time of the transportation operation f'_{ij}, which involves exploring all robots for each assignment.
- Rule 2: In the case of two or more robots provide the minimal completion time f'_{ij}, we choose the robot that has the minimal empty robot arrival time to arrive at the departure machine for the loaded move T_{ij}.

The Heuristic pseudo code is given on algorithm 1:

Algorithm 1. *(HC)* algorithm Pseudocode

```
Function generate_ AR;
Data: Problem inputs (operating range of jobs, processing times, and transfer times).
Result: AR : robot assignement vector
Begin
   AR:= $\emptyset$ /* set of scheduled operations */

   AT := Transport (OM) /* Vector AT is deduced from OM */
   while ($AT \neq \emptyset$)

      rs := Rule_Robot (AT, R) /* Assign a robot to the first operation Tij of vector (AT) according rules 1 & 2.*/
      AR:= Update (AR, $r_s$)  /* Adding $r_s$ to AR */
      AT := AT - $T_{ij}$ /* Subtract $T_{ij}$ from AR */

   end while
end
```

4.4 Genetic Algorithm operators

Crossover:
We used a 1-Point Crossover technique. We randomly selects any crossover point pi (i = 1 to Nj). An offspring vector *(E→OM)* is created by combining *OM* vectors of two parents *(P1→OM & P2→OM)* at crossover point.
The crossover operator should enable inheritance of the number of appearance of each job. After the 1-point crossover we count the appearance number of each job J_i and correct the created offspring to ensure that each job *Ji* appear *nj* times.

Mutation:
The mutation operator that we used is called Simple Exchange Mutation which selects two genes from a chromosome (vector DM) at random and then exchanges their positions. Mutation is applied with small probability P_m since large probability of mutation may lead to loss of the good genes and, as a result, a slowdown of algorithm.

After setting *(E→OM) following crossover and eventually Mutation operations*, *(E→AR)* is set by applying the greedy Heuristic (HC) described in 4.3.

Selection:

- Parent selection for offspring production:

We use a tournament selection method to choose Parent 1 and Parent 2. In each iteration we select Parent P1 and Parent P2 by tournament from two set of K individuals randomly chosen from the population POP. The crossover is then applied on P1 & P2 to create the offspring E.

- Chromosome Selection for replacement

We use a roulette wheel method to select the individual to be replaced Rp. If the fitness of Rp is less than the fitness of E, then Rp is replaced by E on the population (elite selection).

4.5 Hybrid Genetic Algorithm Pseudo

The basic pseudo code of our Hybrid Approach is given on Algorithm 2:

Algorithm 2. *(HGA)* algorithm Pseudocode

```
Function : Hybrid_Genetic_Algorithm (P Instance)
Data: Problem inputs (P Instance).
Result: Best: the Best chromosome
Algorithm parameters:
mni : Maximum number of iterations
mnii : Maximum Number of non-improving iteration to restart
$P_m$ : mutation probability
$P_{New}$ : Percentage of population to be replaced after mnii is reached
$N_{PoP}$ : Population size
Variable:
POP : Chromosome Population
P1, P2, E, R, Best : Chromosomes
ni, nii : Integer
Begin:
    PoP := population_initiale () \*Generation of initialPopulation
    ni:= 0 current number iteration
    nii := 0 \*Number of non improving iteration
    While (ni < mni)
        (P1, P2) : = Selectparent (POP) \*Tournament selection
        E→OM : = Crossover (P1→OM, P2→OM) \*
        If ( Random < Pm)
            Then E→OM := Mutate (E→OM) \*Chromosome mutation with $p_m$ probability
        EndIf
        E→OM :=Repairdeadlock(E→OM) \*Repair OM vector if its sequence is unfeasible.
        E→AR := GreedyHeuristic(E→OM)\* To assign robot to transport Operations
        If  DOUBLE (POP,E) = FAUX \*if E is not a double
            Then R : =Select_Remp (POP) \* Roulette wheel selection.
                If (Fitness (E) > Fitness (R))
                    Then Pop := Replace(E,R, POP)\* Replace chromosome R by E
                         nii :=0
                    Else          nii : =nii+1
                EndIf
        EndIf
        If (nii := mnii)
            Then  POP := Restart (Pop, $P_{New}$) and nii=0 \*Generate randomly a $P_{New}$ chromosome.
        EndIf
        ni : =ni + 1
      EndWhile
Return:
Best \*Return the best population chromosome
End
```

5 NUMERICAL RESULTS

The evaluation of our methods is carried out using a first set of instances from a well Known Benchmark for the problem of a job shop with transport and second set of instances developed by ourselves. The Well-known benchmark is suggested by [23] and has been used by [36]. This instance set encompass one subsets *P1* whose size *(n * m)* is *(6 * 6)*. The second set of instances comprises *15* instances grouped into three subsets *S1*, *S2* and *S3* with respective sizes of *(4 * 4)*, *(6 * 6)*, *(10 * 10)*. For these instances, the jobs routing are randomly generated and the processing times and transfer times are randomly distributed respectively over *[10; 100]* and *[1; 20]*. The evaluations are carried out using respectively *1* robot, *2* robot. For all instances, that encompass more than one robots, the robots are considered as similar.

The experiments were performed using the following set of Parameters: mni : 5000; mnii : 100; Pm : 5%; P_{New} : 10%; N_{PoP} : 100.

In the table 3, we report the results of our Hybrid Genetic Algorithm (HGA) and the MILP[47]. These results are obtained using the *C++* programming language for HGA and *CPLEX 12.6* as LP solver. The programming codes were run on a simple desktop PC running Windows 10 with a Intel Core i5 processor running at 2.40GHz with 8,00GB of memory.

The notations below are used in the numerical results table:

- C_max[1]: The optimal solution found by the execution of MILP for the NWBT JSSP. We set the time limit for each problem instance to 4 Hours. If if the problem could not be solved to optimality when the time limit had been reached then we report the lower Bound and we specify this by (LB).
- *C_max[2]:* the approached solution found by HGA.
- *Dev:* The relative deviation between *C_max[1]* and the *C_max[2]*.

Table 3: Comparison of Numerical results obtained by HGA and MILP.

		With One Robot			With 2 -Robot		
Set Size	Instance	C_max[1]	C_max[2]	Dev%	C_max[1]	C_max[2]	Dev%
P1: (6*6)	D1-d1	94	105	11,70%	68	74	8,82%
	D1-t1	82	86	4,88%	68	71	4,41%
	D2_d1	152	162	6,58%	101 (LB)	125	23,76%
	D3-d1	171	182	6,43%	125 (LB)	147	17,60%
	T2-t1	112	125	11,61%	79 (LB)	100	26,58%
	T3-t0	77	82	6,49%	58 (LB)	72	24,14%
	Tkl.1	104	112	7,69%	77 (LB)	96	24,68%
			Avg Dev	7,91%		avg Dev	18,57%
S1: (4*4)	S1.1	209	209	0,00%	341	209	0,00%
	S1.2	181	181	0,00%	304	181	0,00%
	S1.3	215	215	0,00%	386	215	0,00%
	S1.4	163	165	1,23%	384	164	3,14%
	S1.5	188	190	1,06%	602	190	1,06%
			Avg Dev	0,46%		Avg Dev	0,84%
S2: (6*6)	S1.1	367	396	7,90%	209	379	11,14%
	S1.2	310	334	7,74%	181	339	11,51%
	S1.3	412	439	6,55%	215	435	12,69%
	S1.4	403	446	10,67%	159	436	13,54%
	S1.5	604	655	8,44%	188	652	8,31%
			Avg Dev	6,16%		Avg Dev	10,05%
S3: (10*10)	S3.1	1235 (LB)	1959	58,62%	867 (LB)	1399	61,36%
	S3.2	1308 (LB)	1801	37,69%	908 (LB)	1416	55,95%
	S3.3	1317 (LB)	1930	46,55%	914 (LB)	1398	52,95%
	S3.4	1140 (LB)	1787	56,75%	759 (LB)	1215	60,08%
	S3.5	1042 (LB)	1640	57,39%	881 (LB)	1523	72,87%
			Avg Dev	51,40%		Avg Dev	60,64%

Since there is no benchmark results for the NWBT JSSP, the evaluation of the performance of our Hybrid Genetic Algorithm is done by comparing its results with MILP.

The absolute quality of the results obtained by *HGA* are only compared for small and medium size instances. For large instances, we could not obtain an optimal solution within a reasonable computational time by MILP to compare it with the HGA. From our computational results in Table 3 and over the *34* test instances (Small and Medium Size), we noted that HGA Solution is on average 9,10% larger than MILP results.

6 CONCLUSION

This paper addresses the NWBT JSSP problem with the objective of minimizing the make span. We have modeled the problem by a disjunctive graph and proposed a hybrid method based on a Genetic Algorithm for setting operations sequences, a repair procedure to get feasible schedules and a greedy heuristic to assign robots to transport operations. Since no benchmark tests for the NWBT JSSP problem were available in the literature, we have evaluated our methods by comparing our results firstly with the results obtained by a MILP.

Numerical application show that our Hybrid Genetic Algorithm find a good solutions compared to MILP for a small and large instances and remains more interesting for very large instances in that it allows us to find approximate solutions in reasonable times.

For further research, it would be interesting to ameliorate the solution obtained by our Hybrid Genetic Algorithm by investigating other Heuristics. These heuristics can be Meta –heuristics, such as Artificial Immune Systems, Tabu Search, simulated annealing, or other Hybrid Methods which can be a couple of two Metaheuristics even for Operations sequencing and Robot assignments .

References

[1] A.S. Jain, S. Meeran, Deterministic job-shop scheduling: Past present and future, European Journal of Operational Research 113 (1999) 390-434.

[2] N.G. Hall, C. Sriskandarajah, A survey of machine scheduling problems with blocking and no-wait in process, Operations Research 44 (3) (1996) 510-525.

[3] O. Candar, Machine scheduling problems with blocking and no-wait in process, Working Paper [April-99], Department of Industrial Engineering, Bilkent University, Ankara, Turkey, 1999.

[4] Mascis, D. Pacciarelli, Machine scheduling via alternative graphs, Research Report, RT-DIA-46-2000, Italy, 2000.

[5] A. Mascis, D. Pacciarelli, Job-shop scheduling with blocking and no-wait constraints, European Journal of Operational Research 143 (2002) 498-517.

[6] Y. Mati, N. Rezg, X. Xie, Geometric approach and taboo search for scheduling flexible manufacturing systems, IEEE Transactions on Robotics and Automation 17 (6) (2001) 805-818.

[7] C.A. Brizuela, Y. Zhao, N. Sannomiya, No-wait and blocking job-shops: Challenging problems for GA's, IEEE 0-7803-77-2/ 01 (2001) 2349-2354.

[8] A. Klinkert, Optimization in design and control of automated high-density warehouses, Doctoral Thesis No. 1353, University of Fribourg, Switzerland, 2001.

[9] H. Groflin, A. Klinkert, Local search in job shop scheduling with synchronization and blocking constraints, Internal working paper [04-06], Department of Informatics, University of Fribourg, Switzerland, 2004.

[10] P. Brucker, T. Kampmeyer, Cyclic job shop scheduling problems with blocking,Ann. Oper. Res. 159 (2008) 161–181.

[11] A. AitZai, B. Benmedjdoub, M. Boudhar, A branch and bound and parallel genetic algorithm for the job shop scheduling problem with blocking, Int. J. Oper. Res.14 (3) (2012) 343–365.

[12] H. Gröflin, D.H. Pham, R. Bürgy, The flexible blocking job shop with transfer andset-up times, J. Comb. Optim. 22 (2) (2011) 121–144.

[13] D. Pacciarelli, M. Pranzo, Production scheduling in a steelmaking-continuous casting plant, Computer and Chemical Engineering 28 (2004) 2823–2835.

[14] J. Romero, L. Puigjaner, T. Holczinger, F. Friedler, Scheduling intermediate storage multipurpose batch plants using the S-graph, AIChE J. 50 (2004) 403–417.

[15] R. Zhang, S.J. Song, C. Wu, A hybrid differential evolution algorithm for job shopscheduling problems with expected total tardiness criterion, Appl. Soft Comput.13 (3) (2013) 1448–1458.

[16] T.C. Wong, S.C. Ngan, A comparison of hybrid genetic algorithm and hybridparticle swarm optimization to minimize makespan for assembly job shop,Appl. Soft Comput. 13 (3) (2013) 1391–1399.

[17] J. Hurink ,S. Knust, Tabu search algorithms for job-shop problems with a single transport robot, European Journal of Operational Research 2005;162(1):99–111.

[18] J. Blazewicz,H. Eiselt ,G. Finke,G. Laporte,J. Weglarz, Scheduling tasks and vehicles in a flexible manufacturing system, International Journal of Flexible Manufacturing Systems 1991;4(1):5–16.

[19] U. Bilge,G. Ulusoy, A time window approach to simultaneous scheduling of machines and material handling system in an FMS. Operations Research 1995;43(6):1058–70.

[20] G. Ulusoy, F. Sivrikaya-erifolu, U. Bilge, A genetic algorithm approach to the simultaneous scheduling of machines and automated guided vehicles, Computers \& Operations Research 1997;24(4):335–51.

[21] G. Alvarenga,G. Mateus, G. De Tomi, A genetic and set partitioning two-phase approach for the vehicle routing problem with time windows, Computers \& Operations Research 2007;34(6):1561–84.

[22] M. Soylu,N. O¨ zdemirel, S. Kayaligil, A self-organizing neural network approach for the single AGV routing problem, European Journal of Operational Research 2000;121(1):124–37.

[23] J. Hurink,S. Knust, A tabu search algorithm for scheduling a single robot in a job shop environment, Discrete Applied Mathematics 2002;119(1–2):181–203.

[24] T. Abdelmaguid,A. Nassef, B. Kamal, M. Hassan, A hybrid GA/heuristic approach to the simultaneous scheduling of machines and automated guided vehicles, International Journal of Production Research 2004;42(2):267–81.

[25] P. Lacomme,M. Larabi, N. Tchernev, A disjunctive graph for the job shop with several robots, In: MISTA conference; 2007. p. 285–92.

[26] L. Deroussi, M. Gourgand, N. Tchernev, A simple metaheuristic approach to the simultaneous scheduling of machines and automated guided vehicles, International Journal of Production Research 2008;46(8):2143–64.

[27] H. Gong, L. Tang, Two-machine flow shop scheduling with intermediate transportation under job physical space consideration, Computers \& Operations Research 2011;8(9):1267–74.

[28] M. Mateo, R. Companys, J. Bautista, Resolution of graphs with bounded cycle time for the cyclic hoist scheduling problem, In: 8th International Workshop on Project Management and Scheduling.

[29] A. Caumond, P. Lacomme, A. Moukrim, N. Tchernev, An MILP for scheduling problems in an FMS with one vehicle, European Journal of Operational Research 2009;199(3):706–22.

[30] Y. Crama, J. Van de Klundert, Cyclic scheduling of identical parts in a robotic cell. Operations Research 1997;45(6):952–65.

[31] N. Brauner, G. Finke, V. Lehoux-Lebacque, C. Potts, J. Whitehead, Scheduling of coupled tasks and one machine no wait robotic cells, Computers \& Operations Research 2009;36(2):301–7.

[32] AI. Corréa, A. Langevin, L-M. Rousseau, Scheduling and routing of automated guided vehicles: a hybrid approach. Computers \& Operations Research 2007; 34(6):1688–707, [part special Issue: Odysseus 2003 second international workshop on freight transportation logistics.

[33] E. Stafford, F. Tseng,J. Gupta, Comparative evaluation of MILP flow shop models, Journal of Operations Research Society 2005; 56;88–101.

[34] Nouri, H.E., Driss, O.B., Ghédira, K.: A classification schema for the job shop scheduling problem with transportation resources: state-of-the-art review. Artificial Intelligence Perspectives in Intelligent Systems. AISC, vol. 464, pp. 1–11. Springer, Cham (2016).

[35] Roy, B., Sussmann, B. Les problemes d'ordonnancement avec constraintes disjonctives, Note DS no. 9 bis, SEMA, 1964, Paris.

[36] LARABI, Mohand. Le problème de job-shop avec transport: modélisation et optimisation. 2010. Thèse de doctorat. Université Blaise Pascal-Clermont-Ferrand II.

[37] A Mascis, D Pacciarelli, Job-shop scheduling with blocking and no-wait constraints, European Journal of Operational Research, 2002, Volume 143, Issue 3, Pages 498-517

[40] Dabah A, Bendjoudi A, AitZai A: Hybrid multi-core CPU and GPU-based B&B approaches for the blocking job shop scheduling problem, Journal: Journal of Parallel and Distributed Computing, 2018, Volume 117, Page 73

[41] Nouri H.E., Belkahla Driss O., Ghedira K.Hybrid metaheuristics for scheduling of machines and transport robots in job shop environment, Applied Intelligence, 45 (3) (2016)

[42] Baruwa, O.T., Piera, M.A.: A coloured Petri net-based hybrid heuristic search approach to simultaneous scheduling of machines and automated guided vehicles. Int. J. Prod. Res. 54, 4773–4792 (2016)

[43] Christian Bierwirth. A generalized permutation approach to job shop scheduling with genetic algorithms. Operations- Research-Spektrum, 17(2):87{92, 1995.

[44] Holland J. Genetic algorithms. Scientific American 1992; 267(1):66–72.

[45] Holland, J. (1975). Adaptation in natural and artificial systems. Ann Arbor: University of Michigan Press.

[46] Jones, A., & Rabelo, L. C. (1998). Survey of job shop scheduling techniques. Gaithersburg, MD: NISTIR, National Institute of Standards and Technology.

[47] Louaqad, S & Kamach,O.: Scheduling of blocking and no wait job shop problems in robotic cells. Xème Conférence Internationale : Conception et Production Intégrées, Dec 2015, Tanger, Morocco.

A randomized search procedure combined with simulated annealing for the capacitated location routing problem

A. Lemouari[1]

1. LMAM ,Bât A, University Mohamed Seddik Benyahia
BP 98 18000, JIJEL, ALGERIA

Abstract: In this paper we present a Greedy randomized search procedure (GRASP) combined with the simulated annealing procedure to resolve the capacitated location routing problem, the objective is to determine the total costs combined from fixed cost associated with opening a depot at each potential site, and the total cost of a route includes a fixed cost and the transportation. The proposed method gives best results competitive with others heuristics in literature, able to exceed best known solutions for some instances.

Keywords: greedy randomized search, simulated annealing, capacitated location routing problem.

1 Introduction

The location routing problem (LRP) is fundamental problems of the supply chain management; it can be defined as follows: given a set of customers with known demand and a set of potential depot sites. The objective reached consists to minimizing the sum of the routing costs and fixed costs associated with the selected depots. A special case of the location routing problem is the capacitated location routing problem (CLRP) where capacity on vehicles and depots are imposed, the CLRP well known as NP-hard problems.

Prins et al. gives the following binary integer programming model for the CLRP-problem [4]. Let be an undirected graph where V is a set of nodes, E is a set of edges. The set where I is a subset of m potential depot sites, and J is a subset of n customers. For each edge is associated a cost $_{ij}$, which represents the travelling distance between nodes i and j. Each depot site has a capacity and an opening cost O_i. Each customer $j \in J$ has a demand which must be fulfilled by a single vehicle. A set k of identical vehicles with capacity Q is available. Each vehicle performs a single route, and when used, incurs a fixed cost F.
The main objective is to minimize the following function

The following constraints are imposed for the problem:

- Each route must start and finish at the same depot; and each vehicle performs a single route;
- each customer is visited exactly once by a single route;
- the sum of the demands of the customers visited by each route must not exceed the vehicle capacity;
- the sum of the demands of the customers assigned to each depot must not exceed its corresponding capacity;
- connections between depots are not allowed.
- Each route must start and terminate at the same depot, and its total load must not exceed vehicle capacity.
- The total load of the routes assigned to a depot must fit the capacity of the depot.

Prins et al. [4] solved the CLRP with a Greedy Randomised Adaptive Search Procedure (GRASP) followed by a path relinking algorithm and Duhamel et al. [3] proposed a GRASP hybridized with Evolutionary local search procedure. In [5] authors propose a two-phase hybrid heuristic algorithm to solve the CLRP, in the proposed hybrid heuristic algorithm, after a Construction phase a modified granular tabu search, with different diversification strategies, is applied during the Improvement phase. The algorithm is able to provide good solutions within very fast run-time. In the rest of this summary, we outline our proposal.

2 Proposed Algorithm

Construction phase

In this phase, we propose a procedure to construct an initial feasible solution. The goal is to open a set of potential facilities that cover all clients with a minimal cost. Three selection models were randomly used, the first was proposed by Prins et al. the other two are a simplified version of the first

1. —— 2. ——— 3. CS

Where CS_i is the cost of selected depot i, : opened cost of depot i, : the capacity depot, and is the distance between depot i and customer j.

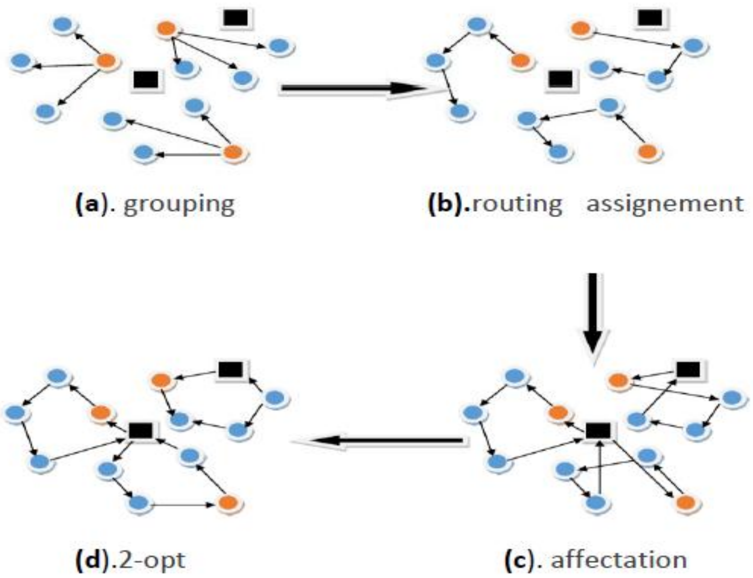


Fig. 1 construction phase.

After having classified all deposits and selecting the ones that will be opened, the solution is built according to the following steps (Fig. 1):

- **Grouping:** based on a greedy algorithm used for the K-center problem [6].
- **Assignment:** The second step is to determine routes.
- **Affectation:** Assign tours to depots select.
- **2-opt:** Application of the 2-opt operator in order to escape the crossings.

Improvement phase

Once the solution is built an improvement phase based on the simulated annealing heuristic **SA()** inspired from [2], will succeed. The difference is that in our implementation the algorithm and for the same solution is called the number of open facilities. The following movements were used to determine the best neighbors: permutation, moving from and **CROSS-exchange** operator.

After the complete solution is thus reconstructed, a local search procedure **Localsearch (s)** is used, which takes into account inter-depot movements.

3 Preliminary Tests

The algorithm has been implemented in JAVA Net Beans; the computational experiments have been performed on an Intel Core™ i7-2600 CPU (3.40 GHz). The preliminary performance test of the proposed algorithm has been evaluated by considering data set proposed by Prins et al. [1], contains 30 instances with capacity constraints on both the routes and the depots. The number m of potential depots is either 5 or 10, and the number of customers' n numbered in [20, 50, 100, 200].

The customers and the depots correspond to random points in the plane. The traveling costs are calculated as the corresponding Euclidean distances, multiplied by 100 and rounded up to the next integer. The vehicle capacity Q is either 70 or 150, and the demands of the customers are uniformly random distributed in the interval [11, 20]. Our preliminary tests give, for instance types of 100 customers 5 depots and 100 customers 10 depots, the following results

100 Customers

#	**BKS**	GRASP	GRASPXELS	SALRP*	2-HGTS	SA-GRASP	RC	LC	D/R
100/5/1a	275419	279437	276960	275419	276186	278512	145622	132890	3/24
100/5/1b	213615	216159	215854	213615	214892	217831	84941	132890	3/12
100/5/2a	193671	199520	194267	193671	194625	195214	92968	102246	2/24
100/5/2b	157150	159550	157375	157150	157319	157946	55700	102246	2/11
100/5/3a	200079	203999	200345	200079	201086	**200003***	112857	87146	2/24
100/5/3b	152441	154596	152528	152441	153663	153671	65384	88287	2/11
100/10/1a	287983	323171	301418	287983	289755	**287306***	132375	154931	3/25
100/10/1b	231763	271477	269594	231763	238002	232565	84242	148313	3/12
100/10/2a	243590	254087	243778	243590	245768	246601	103211	143390	3/25
100/10/2b	203988	206555	203988	203988	204252	**203851***	60461	143390	3/11
100/10/3a	250882	270826	253511	250882	254716	251066	120075	130991	3/25
100/10/3b	204317	216173	205087	204317	205837	**200928***	69937	130991	3/11

BKS : Best known Solution, **RC:** Routing Cost, **LC**: Localisation Cost, **D/R**: Opened Depot / Routes

References

[1] C. Prins, C. Prodhon, R. Wolfler-Calvo (2004). Nouveaux algorithmes pour le problème de localisation et routage sous contraintes de capacite´ . In: MOSIM, Nantes, France, vol. 4, 1115–1122.

[2] V.F. Yu, S. Lin, W. Lee, C. Ting (2010). A simulated annealing heuristic for the capacitated location routing problem. Computers and Industrial Engineering. 58(2), 288–299.

[3] C. Duhamel, P. Lacomme, C. Prins, C. Prodhon (2010). A GRASPxELS approach for the capacitated location-routing problem. Computers and Operations Research, 37(11),1912–1923.

[4] C. Prins, C. Prodhon, R. Wolfler-Calvo(2006). Solving the capacitated location-routing problem by a GRASP complem ented by a learning process and a path relinking. 4OR, 4(3), 221–238.

[5] J.W. Escobar, R. Linfati, P. Toth (2013). A two-phase hybrid heuristic algorithm for the capacitated location-routing problem. Computer & operations research, 40(1), 70-79.

[6] T. Gonzalez (1985). Clustering to minimize the maximum inter-cluster distance. Theoretical Computer Science., 38, 293–306.

Design and Parallel Implementation of the H264 application on Heterogeneous Architectures

C. Adda[1] and A. Benyamina[2]

1. University of Tiaret Ibn khaldoun, LIM laboratory, Tiaret Algeria
chahrazedadda@gmail.com

2. University of Oran1 Ahmed Ben Bella, LAPECI laboratory, Oran Algeria
banyanabou@yahoo.fr

Abstract: The H.264 is an emerging video coding standard, which aims at compressing high-quality video contents at low-bit rates. While the new encoding and decoding processes are similar to many previous standards, the new standard includes a number of new features and thus requires much more computation than most existing standards do. The complexity of H.264 standard poses a large amount of challenges to implementing the encoder/decoder in real-time requiring large amount of processing resources. This paper presents the design and analysis of the H.264 decoder implemented on a heterogeneous architecture (multi-CPUs/multi-GPUs). A model-driven approach is adopted by using the standard MARTE profile of UML. Our approach is based on hybrid partitioning that combines both functional and data partitioning which is applied to find the most suitable processors (CPU or GPU) regarding the execution time. We claim that our approach allows giving a better performance, which is crucial when implemented in modern complex systems.

Keywords : General-Purpose Graphics Processing Unit (GPGPU), Multimedia, H.264/AVC decoder, Parallel Processing, Functional Partitioning, Data Partitioning.

1 Introduction

In recent years, the performance improvement of Graphics Processing Unit (GPU) is remarkable and GPUs are becoming attractive as accelerators for heavy tasks. GPUs are now commonly used as co-processors in many embedded systems to accelerate general-purpose applications. They are particularly capable of executing data-parallel applications, due to their highly multithreaded architecture and high-bandwidth memory. Various embedded system domains can benefit high performance and better energy efficiency from utilizing GPUs. For example, GPUs can efficiently perform matrix operations such as factorization on large data sets and multidimensional FFTs and convolutions. Such operations are often seen in many embedded applications including signal processing, imaging and video processing. By leveraging new programming models, such as CUDA [1] and OpenCL [2], programmers can effectively develop highly data-parallel kernels to execute such applications on GPUs.

By integrating heterogeneous processing elements with different performance characteristics in the same system, heterogeneous CPU/GPU architectures are expected to provide more flexibility for better performance compared to homogeneous systems. Execution time and energy consumption are imperative performance metric that needs to be optimized in most embedded systems. In order to minimize execution time and energy consumption for running a set of workloads, the step that partitioning computations to processing elements is critical. In this paper, we consider the partitioning of workload problem in a heterogeneous system containing multiple CPUs and GPUs. Our goal is to minimize the execution time.

Embedded applications that we are used to process are generally complex such as MPEG, H263, and H264.......etc etc encoders. This type of application is characterized by parts of different types, a party is treated regularly and the other irregularly. The latter represents intensive computing. For this type of application two types of parallel processing are applied, regular processing (task Parallelism) and irregular processing (data Parallelism). Furthermore, video coding standards like H.264/AVC [3] and HEVC [4] are adopting complex algorithms like context-adaptive binary arithmetic coding (CABAC) and variable length coding (CAVLC) in order to achieve better compression and thus lower transmission bitrates for high resolution video sequences. The additional complexity of these algorithms has a major impact by increasing execution time and energy consumption.

In our research, we intend to solve the problem of high complexity of the H.264 decoder using parallelization on multicore embedded processors and on graphical processors. Video resolutions are increasing rapidly, which require more processing time and consequently more energy consumption. Many solutions based on parallel execution exist ranging from macroblocks (fine-grain) till groups of pictures (coarse-grain) parallel decoding. Macroblock parallel decoding is highly scalable since many macroblocks can be processed in parallel. However, dependencies and huge overheads are created as a result of communication and synchronization between macroblocks. Parallel decoding of groups of pictures require large memories for high definition video sequences. In addition, they have a lower scalability than macroblock decoding because of the limited number of groups of frames that can be decoded in parallel. Our solution is based on the H264 video decoder design taking into account the different parallelism and heterogeneous multiprocessor architecture CPUs/GPUs.

Our main contribution in this paper is to propose a new approach based on modeling analyze the computational requirements of H.264 decoder and implement H.264 decoder on heterogeneous architecture (multi-CPUs/multi-GPUs) in order to minimize execution time. Our approach utilizes parallel processing techniques such as workload partitioning.

The remainder of the paper is organized as follows. In Section 2, we present the related work concerning H.264 parallel optimizations. In Section 3, we describe our approach for parallel execution of macroblock rows of the H.264 decoder. In Section 4, we present the experimental results for execution time on CPUs and GPUs using a simulator for multicore processors. Final conclusion and future work are given in Section 5.

2 Related Works

Ever since the H.264/AVC standard [3] was published in 2003, researchers started to solve the high complexity issue of the new standard mainly using parallelism. Several modifications were suggested for the H.264 encoders and decoders in order to improve the performance in terms of execution time and memory usage. Parallel decoding techniques of H.264 exist from the highest level, which is the group of frames or pictures (GOP), the coarse-grain level, till the lowest level, which is the block inside a macroblock, the fine-grain level. Kannangara [5] reduced the complexity of the H.264 decoder (19-65%) by predicting the SKIP macroblocks using an estimation based on a Lagrangian rate-distortion cost function. Gurhanli [6] suggested a parallel approach by decoding independent groups of frames on different cores. The speedup is conditioned with the modification of the encoder in order to omit the start-code scanner process. Any modification to the encoder will require a long process for modifying the H.264 specification in order to be compliant with the standard. The exclusion of previously encoded video sequences is also an effect for modifying the H.264 encoder. Nishihara [7] proposed a load balancing mechanism among cores where partitions sizes are adjusted during runtime. He also reduced the memory access contention based on execution time prediction. Among frame-level and MB-level parallelization, the 3D-wave technique proposed by Azevedo [8] decodes independent MBs in parallel on different cores. A good scalability is proved for HD resolutions where macroblocks are scanned in zigzag mode and decode independent macroblocks in parallel. Chong [9] added a pre-parsing stage in order to resolve control dependencies for MB-level parallelization. Van Der Tol [10] mapped video sequences data over multiple processors providing better performance over functional parallelization. He groups macroblocks in a way that minimal dependency between cores is required. Horowitz [11] compared different H.264 implementations including FFmpeg [17] and the H.264 reference software JM [18]. He also analyzed the complexity of the H.264 decoder subsystems. Sihn [12] proposed a multicore pipeline for the deblocking filter based on the group of pictures data level partitioning. He also suggested software memory throttling and fair load balancing techniques in order to improve multicore processors performance when several cores are used. [13] Proposes to decode the lines of the macroblocks in parallel on a certain number of cores of the processors, the dependencies between macroblocks are ignored. [16] Implements H264 decoder on FPGA architecture. In our research we optimize the H.264 decoder knowing that our approach can be also applied to the H.264

encoder. We focus on improving the efficiency of the H.264 decoder using heterogeneous processors CPU/GPU. We model the H264 / AVC decoder with the recent standard MARTE profile [14] taking into account the different parallelism (component parallelism and data parallelism) and the data dependency between macroblocks. We map our implementation on CPU and GPU processors. Execution time implementation is calculated using simulated execution time. We further implement an OpenCL [2] version of our parallel H.264 implementation. Simulation experiments on heterogeneous processors are conducted using a CPU-GPU simulation Multi2Sim [21].

3 Heterogeneous System Architecture (HSA)

The Generic graphics processors or GPGPU (General Purpose GPU) have evolved in such a way that they can now perform parallel calculations for a wide range of applications. However, programming these devices at the same time as the CPU present on the same chip constitutes a major difficulty. A new architectural concept, The HSA (Heterogeneous System Architecture), paves the way for greater fluidity in the development of heterogeneous code.

The GPUs (Graphics Processing Unit) have passed in recent years from the status of purely graphic accelerators to that of generic parallel processors, Supported by standard APIs and tools such as OpenCL and DirectCompute. Despite this promising start, there are still many obstacles to the existence of an environment using the GPU in a manner as transparent as the CPU (Central Processing Unit, General processor) for current tasks of programming. The CPU and the GPU, in particular, manage different memory spaces, and the hardware is not virtualized. The HSA (Heterogeneous System Architecture) architecture eliminates these obstacles, So that the programmer can exploit the parallel processor contained in the GPU as a coprocessor of the same level as the traditional multi-threaded CPU.

HSA is actually a software layer that provides a unified view of the fundamental processing elements, and allows the programmer to write applications that smoothly integrate CPUs and GPUs while benefiting from the best characteristics of each. The underlying strategy is to create a unified programming platform that serves as a foundation for the deployment of languages, Frameworks and applications that exploit parallelism. More specifically, HSA intends to break the programmability barrier CPU/GPU, Reduce communication latency CPU / GPU, Open the platform to a wider range of applications by accepting existing programming models, and prepare the reception of new processing units in addition to CPU and GPU [19].

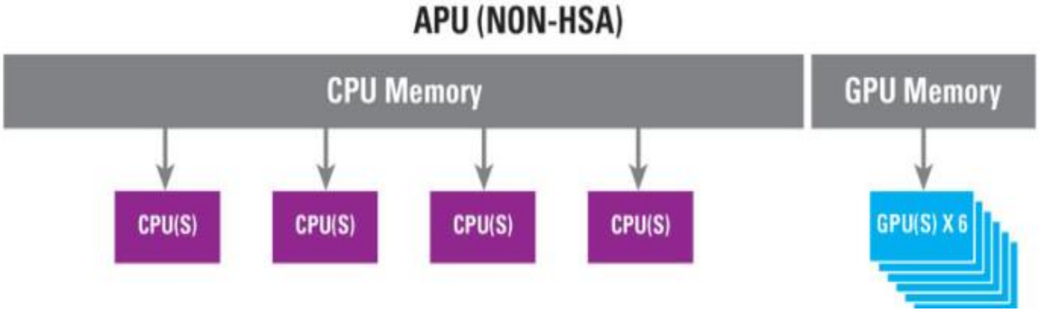


Figure 1: Architecture HAS

New class processors known as Accelerated Processing Unit or APU in a grate CPUs and GPUs in the same computer chip [20], two different processor units working together like a human brain is a enable by heterogeneous system architecture or HSA. “Figure1”. Two sides of AMD API (CPU and GPU) share the same system memory known as heterogeneous Uniform Memory Access or hUMA1 via cache coherent views. Advantages include an easier programming model and less copying of data between separate memory pools.

In our work we are interested by news type of architecture such as heterogeneous architecture GPGPU (multi-CPUs/multi-GPUs) based on Heterogeneous System Architecture (HSA) in order to benefit from advantages of HSA.

4 An Overview of the H.264 Standard

H.264/ AVC [22] video compression standard takes advantage from spatial and temporal redundancy in a video sequence. Therefore, it defines various prediction modes to predict each macroblock depending on its texture properties.

In encoder processing, residual macroblocks consists of difference between original macroblocks and the corresponding predicted one. Residual is the final data organized in bitstream.

Decoder is responsible to reconstruct a video sequence from the compressed data created by encoder. As shown in Figure 1, first step is entropy decoding. It receives the compressed bitstream to reconstruct video parameters and residual coefficients. Then, two primary paths are considered in decoder process. First one is the decoding of residual macroblocks by inverse quantization and inverse transform. Second path is the generation of the predicted macroblocks according to prediction mode fixed by encoder. The addition of the outputs of these two paths is the reconstruct macroblock. A deblocking filter is then applied to have a better video quality. For more details, following sub-sections describe every module of decoder.

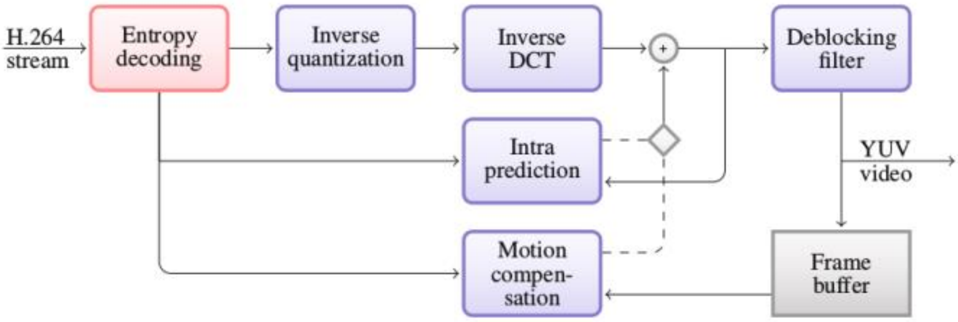


Figure 2: H.264/AVC decoder

4.1 Elements of a Video Sequence

H.264 is a block-based coder/decoder (codec), meaning that each frame is divided into small square blocks called macroblocks (MBs). The coding tools / kernels are applied to MBs rather than to whole frames, thereby reducing the computational complexity and improving the accuracy of motion prediction. Figure 3 depicts a generic view of the data elements in a video sequence. It starts with the sequence of frames that comprise the whole video. Several frames can form a Group of Pictures (GOP), which is an independent set of frames. Each frame can be composed of independent sections called slices, and slices ones, in turn, consist of Mbs. Each MB can be further divided into sub-blocks, which in turn, consist of pixels.

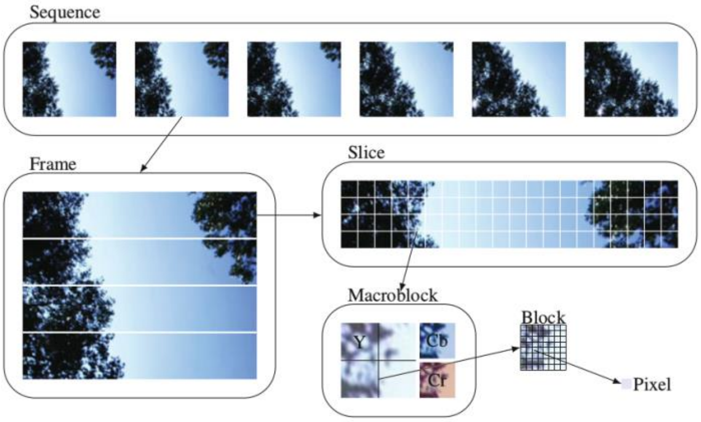


Figure 3: Elements of a video sequence

A MB consists of separated blocks for luma (denoted by Y) and chroma signals (denoted by Cb and Cr). A pre-processing step has to be applied to convert video from a different color component format (such as red-green-blue, RGB) to the Y Cb Cr color model. Chroma sub-sampling is applied to reduce the amount of color information, since the human eye is more sensitive to brightness (Y) than to color (Cb and Cr) [23]. The most common color structure is denoted by 4:2:0 in which the chroma signals (Cb and Cr) are sub-sampled by 2 in both dimensions. As a result, in H.264, as in most MPEG and ITU-T video codecs, each MB typically consists of one 16 × 16 luma block and two 8 × 8 chroma blocks.

4.2 Frame Types

H.264 defines three main types of frames: I-, P-, and B frames. An I-frame uses intra-prediction and is independent of other frames. In intra-prediction, each MB is predicted based on adjacent blocks from the same frame. A P-frame (Predicted frame) uses motion estimation as well as intra-prediction and depends on one or more previous frames, which can be either I-, P- or B-frames. Motion estimation is used to exploit temporal correlation between frames. Finally, B-frames (Bidirectionally predicted frames) use bidirectional motion estimation and can depend on previous frames as well as future frames [24].

Figure 4 illustrates a typical I-P-B-B (first an I-frame, then two B-frames between P-frames) sequence. The arrows indicate the dependencies between frames caused by motion estimation. In order to ensure that a reference frame is decoded before the frames that depend on it, and because B-frames can depend on future frames, the decoding order (the order in which frames are stored in the bitstream) differs from the display order. Thus a reordering step is necessary before the frames can be displayed, adding to the complexity of H.264 decoding.

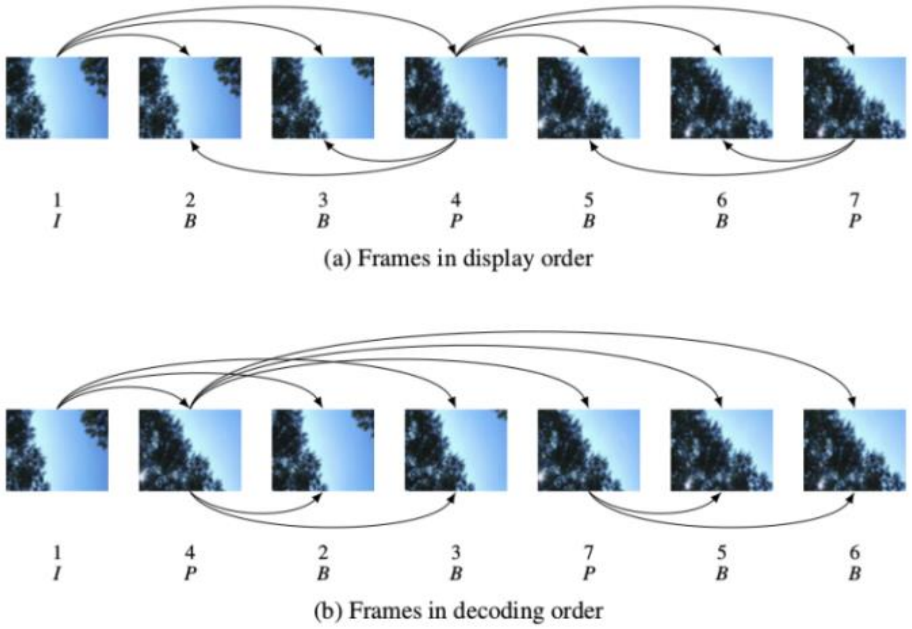


Figure 4: Types of frames and display order versus decoding order

4.3 H.264 Decoding Tools

The H.264 standard has many decoding tools each one with several options. Here we can only briefly mention the key features.

4.3.1 Entropy decoding

After decoding Network Abstraction Layer (NAL) parameters, the data elements are entropy decoded by two ways: Context-based Adaptive Variable Length Decoding (CAVLD) or a binary arithmetic coder (CABAC) which achieves higher compression and Exp-Golomb.

Exp-Golomb: is used for others syntaxes elements such as prediction mode and quantization parameter.

CAVLD is more time consuming than Exp-Golomb [27]. It is used to reconstruct and to reorder data on 4x4 block of 16 integers. By the mean of standard code tables, each 4x4 block is decoded into five syntax elements: Coefftoken, Sign, Level, TotalZeros, and Run [22][23].CAVLDis easier to implement than CABAC.

4.3.2 Inverse quantization

CAVLD output is a residual quantified macroblock. Following step is inverse quantization to produce a set of coefficients () (W_{ij}). Since quantization is a losing information step, inverse quantization reconstructs data. It is multiplication operation as described in equation 1, where $Z_{ij}$is inverse quantization input, W_{ij} is its output and $Q_{step}$is a quantization factor given by standard according to $Q_p$value.

Q_p Is the quantization parameter fixed by encoder. It is decoded from the bitstream using Exp-Golomb codes.

$$W_{ij}=Z_{ij}\cdot Q_{step} \qquad (1)$$

In order to manipulate only integer value in transform step, H.264 standard have postponed real multiplication operation from transform to quantization [23].Details of this operation is given in inverse transform sub section. The final inverse quantization equation given by standard is described by equation 2, whereV_{ij} is the rescaling factor defined by the standard.

$$W_{ij}=Z_{ij}\cdot V_{ij}\cdot 2^{\mathrm{floor}(Q_p/6)} \qquad (2)$$

To implement this equation, a number of shifts equal to "$\mathrm{floor}(Q_p/6)$" was used instead of arithmetic multiplication. Shift operation is less time consuming than multiplication operation.

4.3.3 Inverse transform

In previous video coding standards, Inverse Discrete Cosine Transform (DCT) was used. Inverse transform step is applied for each 4x4 block. For 16x16 Intra prediction modes, a suppliant Hadamard transform is adding for DC Coefficients. Most of the energy is concentrated in the DC coefficients for a 16x16 intra coded macroblock. This extra transform helps to de-correlate the DC coefficients to take advantage of the correlation among coefficients. As shown in Figure 5, DC coefficients of each 4x4 block are assembling in a matrix to applied inverse Hadamard transform given by equation 4. An inverse DC quantization is also applied on DC matrix.

$$R=F^{T}(T\otimes E)F$$

$$\begin{bmatrix} 1 & 1 & 1 & 1/2 \\ 1 & 1/2 & -1 & -1 \\ 1 & -1/2 & -1 & 1 \\ 1 & -1 & 1 & -1/2 \end{bmatrix} \left(\begin{bmatrix} T_{00} & T_{01} & T_{02} & T_{03} \\ T_{10} & T_{11} & T_{12} & T_{13} \\ T_{20} & T_{21} & T_{22} & T_{23} \\ T_{30} & T_{31} & T_{32} & T_{33} \end{bmatrix} \otimes \begin{bmatrix} a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \\ a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \end{bmatrix} \right) \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1/2 & -1/2 & -1 \\ 1 & -1 & -1 & 1 \\ 1/2 & -1 & 1 & -1/2 \end{bmatrix} \quad (3)$$

With $a=\frac{1}{2}$ and $b=\sqrt{\frac{2}{5}}$

$$R_{DCLuma} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \begin{bmatrix} T_{DC0} & T_{DC1} & T_{DC2} & T_{DC3} \\ T_{DC4} & T_{DC5} & T_{DC6} & T_{DC7} \\ T_{DC8} & T_{DC9} & T_{DC10} & T_{DC11} \\ T_{DC12} & T_{DC13} & T_{DC14} & T_{DC15} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \quad (4)$$

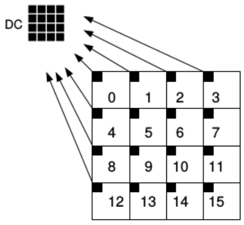


Figure 5: DC coefficients positions in a macroblock

4.3.4 Inverse prediction

Because of redundancy in video sequence, H.264/AVC standard is based on two principal prediction modes [22]. Temporal resemblance between frames is treated as inter prediction. Spatial resemblance in same frame is treated as intra prediction. In LETI decoder, first frame of a sequence is necessarily Intra (4x4 or 16x16) coded because it hasn't reference frame. For next P frames, each macroblock can be coded intra (4x4 or 16x16) or inter prediction.

The 4x4 intra prediction modes are suitable for significant details within a frame. Each 4x4 block is predicted independently from spatially neighboring coefficients. One of nine prediction modes illustrated by Figure 6 [23] is used. According to adjacent block availability, modes can be applied or not. Vertical prediction mode (called also Mode 0) cannot be applied only if top neighboring block at least is available, because this mode copies pixels above the 4x4 block as indicated in Figure 6. For horizontal prediction mode (Mode 1), the pixels to the left of the 4x4 block are copied horizontally if available. Adjacent pixels availability is not necessary to perform DC prediction mode (mode 2). The remaining 6 modes are diagonal prediction modes. They use defined equation to privilege specified direction. Directional modes are suited but they entail additional complexity in the decoding process [25].

The 16x16 intra predictions is characterized by four prediction modes: horizontal mode, vertical mode, DC mode and planer mode. Except of planer mode, all modes have respectively the same propriety of 4x4 modes but they are applied on a 16x16 macroblock. In planer mode, a curve fitting equation is used to form a prediction block having a brightness and slope in the horizontal and vertical directions that approximately matches the neighboring pixels. After statistic work [27], planer mode has been eliminated from LETI encoder because of its supplementary incising complexity relative to its video quality contribution.

In inter prediction case; motion vector is first extracted from bitstream. Then, motion compensation module is applied. It consists of adding motion vector coordinates to corresponding block in reference frame. Result is reconstructed block. Block size can change from one motion vector to other. Different block sizes are supported in H.264/AVC standard, as shown in Figure 7. In LETI decoder only one frame reference is applied and smaller block size for motion vector is 8x8[27].

The data obtained from the intra or inter prediction is added to the inverse transformed residual coefficients. This sum is copied to the decoded buffer which is used as an input for deblocking filter step.

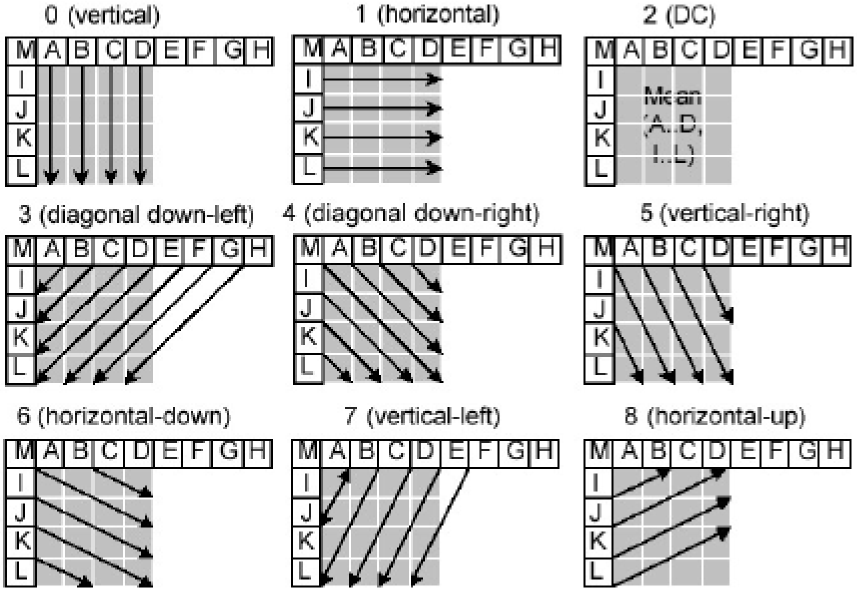


Figure 6: 4x4 Intra prediction modes

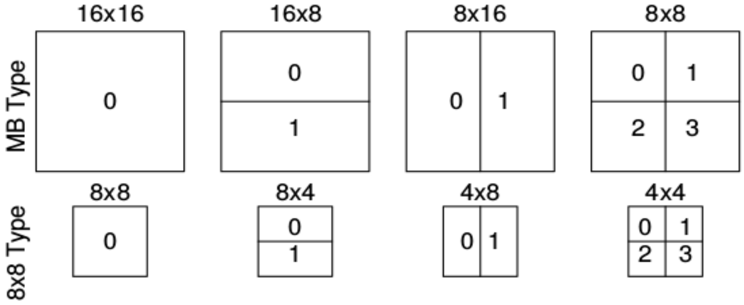


Figure 7: Inter prediction block size type

4.3.5 Deblocking filter

The deblocking filter performs in-loop filtering to reduce blocking artifacts created by image partitioning and quantization. After inverse quantization and inverse transform, the deblocking filter compares the edge values of each 4x4 block with its adjacent block to select the level of filtering. In LETI decoder, Strong or Standard filter is selected according to the block edge, macroblock position in frame and prediction mode. The design algorithm of deblocking filter used in this work is shown with details in [28].

5 H.264 Implementation

5.1 H264 Decoder Partitioning

The H264 decoder process is modeled with the UML / MARTE specification, as shown in the figure 8.Where it's divided into five main functional parts: Entropy Decoder (ED), Inverse Quantization (INVQ), Inverse Transform (IDCT), Inverse prediction (INV-Pred) and and Deblocking Filter (DF). After decoding Network Abstraction Layer (NAL) parameters, the data elements are entropy decoded, that is divided into 3 components: Exp-Golomb is used to extract syntax elements such as the prediction mode and the quantization parameter, CAVLD is used to reconstruct and to reorder data on 4x4 blocks of 16 integers. Incoming maclocks are analyzed with an inter-prediction and intra-prediction in order to increase the coding efficiency by finding redundant information. In the intra-prediction, it determine predicted pixels according to the prediction mode and the macroblock position within a frame (MBX, MBY). Neighboring pixels are generated by a suppliant component called “Neighboring pixels”. In the intra-prediction, macroblocks are decoded by a vector, called motion vector, the component Intra/Inter selects the right prediction. The Sub-component consists in subtracting the redundant information from the initial frame. The component Inverse quantization input is 16 coefficients of a block seized 4x4 the CAVLC outputs, component Inverse Transform used Inverse Discrete Cosine Transform (DCT) Inverse transform step is applied for each 4x4 block. Deblocking filter is executed at the end of the decoding process in order to reduce the edging effect between macroblock borders 4X4 after adding the coefficients predicted and transformed by the component addition.

Whereas components, Inverse Quantization, Inverse Transform and Deblocking filter correspond contain intensive data-parallel computations. The Repetitive Structure Modeling (RSM) package of MARTE offers suitable concepts to describe such computations.

5.2 H264 Decoder Functional Partitioning

According the MARTE model in figure8, the video decoding application is composed of six tasks called Decoding Exp-Golomb (Exponential-Golom), MB-Header, Context-based Adaptive Variable Length Decoding (CAVLD), Inverse Quantization (INVQ), Inverse Transform (INVT), Inverse prediction (INV.Pred) and Deblocking filter (DBfilter). As shown in figure 9.

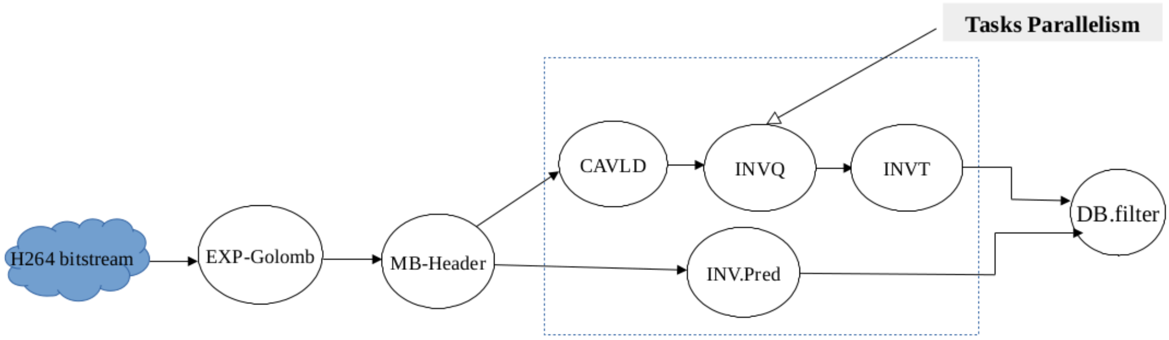


Figure 9: Inter prediction block size type

Exp-Golomb and MB-Header are running in the first place and sequentially because of data dependency. CAVLD, Inverse quantization and inverse transform are running sequentially because of data dependency. Inverse prediction task can be done in parallel with CAVLD, inverse quantization and inverse transform (task Parallelism). Inverse prediction and inverse transform tasks should be completed before running DBfilter.

5.3 H264 Decoder Data Partitioning

5.3.1 Low-level parallelization: decode macroblocks in parallel

In H.264, there are 3 types of macroblocks: I-, P-, and B frames. An I-MB uses intra-prediction., each MB is predicted based on adjacent blocks from the same slice of frame. A P-MB (Predicted macroblock) uses motion estimation as well as intra-prediction and depends on one or more macroblocks from previously decoded frames. Motion estimation is used to exploit temporal correlation between frames. Finally, B-MB (Bidirectionally predicted macroblocks) uses bidirectional motion estimation and can depend on previous frames as well as future frames.

In our work we use the frame sequence I-, P-, and B frames, for more details see section 4.2. Each frame is subdivided into 16x16 or 4x4macroblocks. For each frame sequence we have a single frame of type Ï, we start with coded I-macroblocks sequentially then P-macroblocks and B-macroblocks are coded in parallel. As shown in figure 10.

5.4 Implementation of H264 Decoder on CPUs / GPUs

In our work we are interested by news type of architecture such as heterogeneous architecture GPGPU (multi-CPUs/multi-GPUs) based on Heterogeneous System Architecture (HSA). In HSA, the CPU processor and GPU processor run together and in the same level, data access for the GPU is direct via shared heterogeneous memory.

Exp-Golomb and MB_Header tasks are executed in the first and sequentially because of data dependency on CPU. These two tasks are executed in the same to cancel the data transfer time.

CAVLD, Inverse quantization and inverse transform are running sequentially because of data dependency. These tasks are executed on GPU (first GPU) because this latter contain intensive data-parallel computings and The Repetitive Structure) in order to benefit from advantages of GPU (parallel computing).

Inverse prediction task can be done in parallel with C AVLD, inverse quantization and inverse transform (task Parallelism). This task is executed on GPU (second GPU). The I-macroblocks are coded sequentially on a GPU core, P and B macroblocks are coded in parallel on other GPU cores.

Inverse prediction and inverse transform tasks should be completed before running DBfilter. So DBfilter can be run on either GPU1 or GPU2.

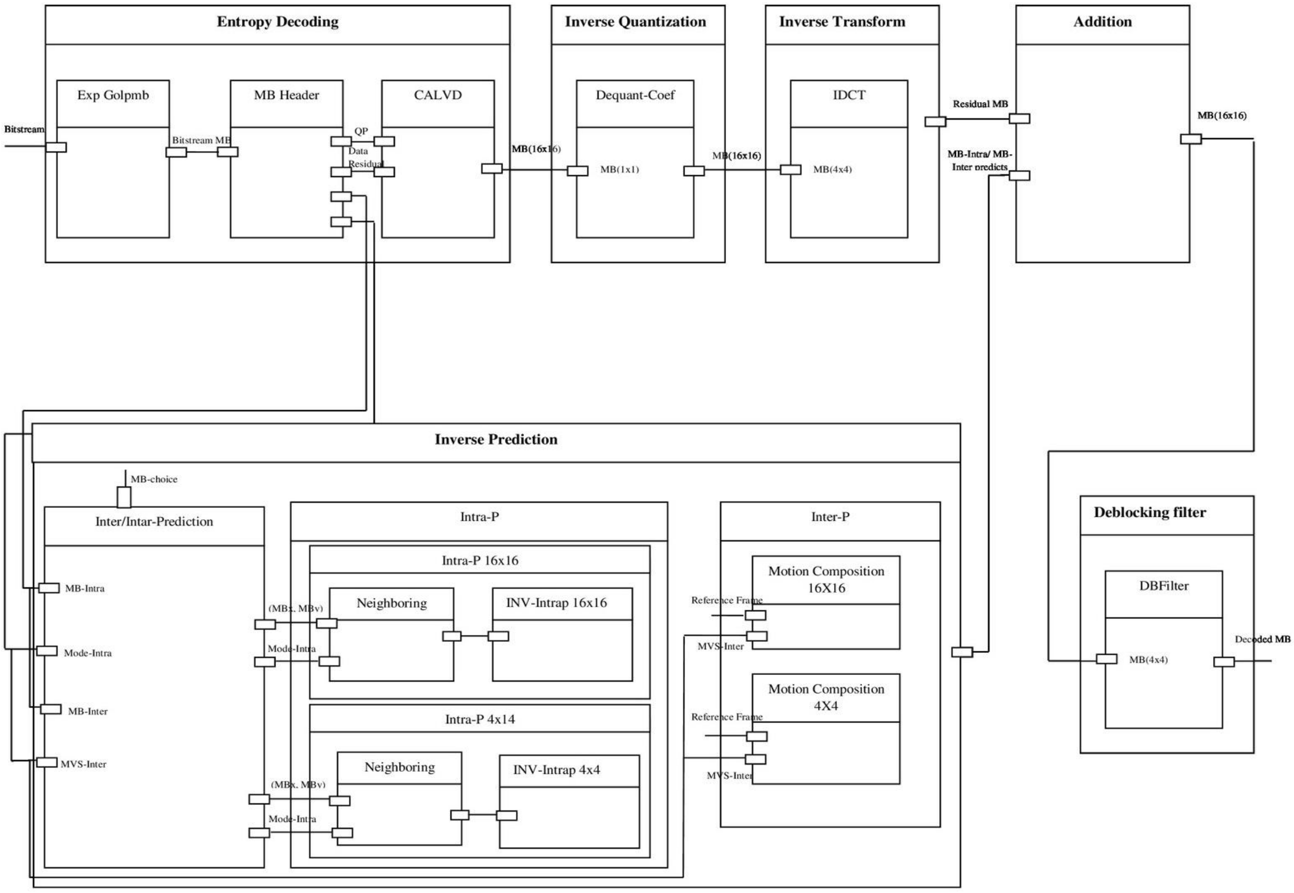


Figure 8: Video application modeling based on the H.264 decoder

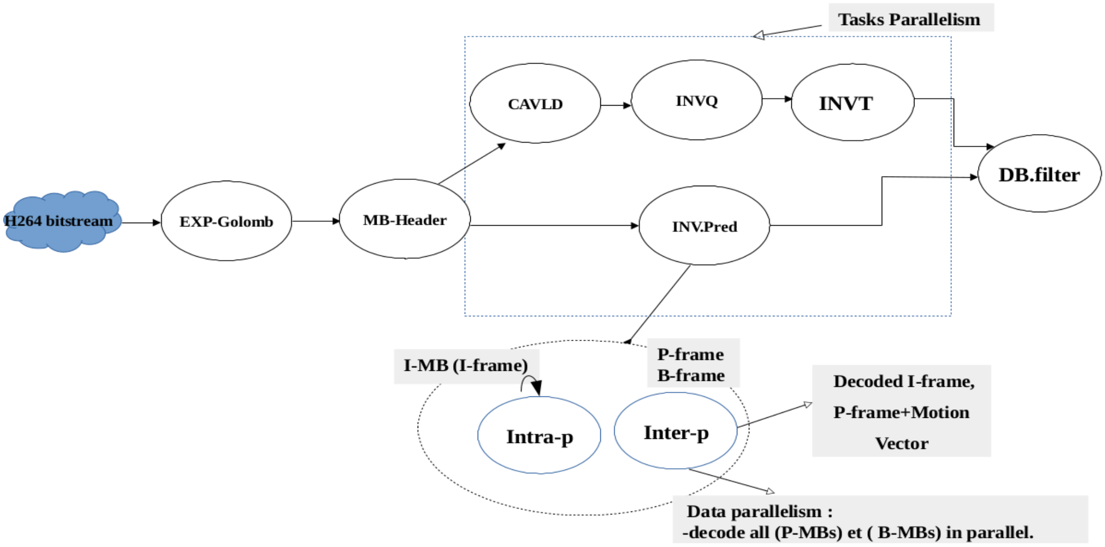


Figure 10: H.264 decoder task graph with Data parallelism

6 Exprimental and Results

6.1 Simulations

The H.264 reference software, JM [18], is an open source implementation used as a reference implementation for the H.264 standards. In our research, we modified the JM [18] source code of the H.264 decoder in order to decode macroblocks in parallel (P et B) and I sequentially, partition the source code into sub-code and run them on the different processor using the PThread library in C programming language. Our H.264 implementation is executed in real HSA architecture by emulator Multi2Sim [21], a cycle-accurate simulator for multicore x86 and graphics processors. Cache and memory configurations comply with common x86 processors that are available nowadays in many Intel [32] or AMD [29] processor chips. Each core has a private L1 cache of 512 KB and All other cores have a shared L2 cache of 2 MB and shared memory between the two processors CPU and GPU. We simulate the execution of our parallel H.264 decoder using 2 GPU multicore (AMD Evergreen) and CPU multicore (x86). We perform simulation experiments of the H.264 OpenCL version on the AMD Evergreen GPU family with the configurations of the AMD Radeon 5870 GPU [31].We gather statistics using 15 video sequences with HD resolution is performed for the H.264 decoding process of 60 frames for each video sequence.

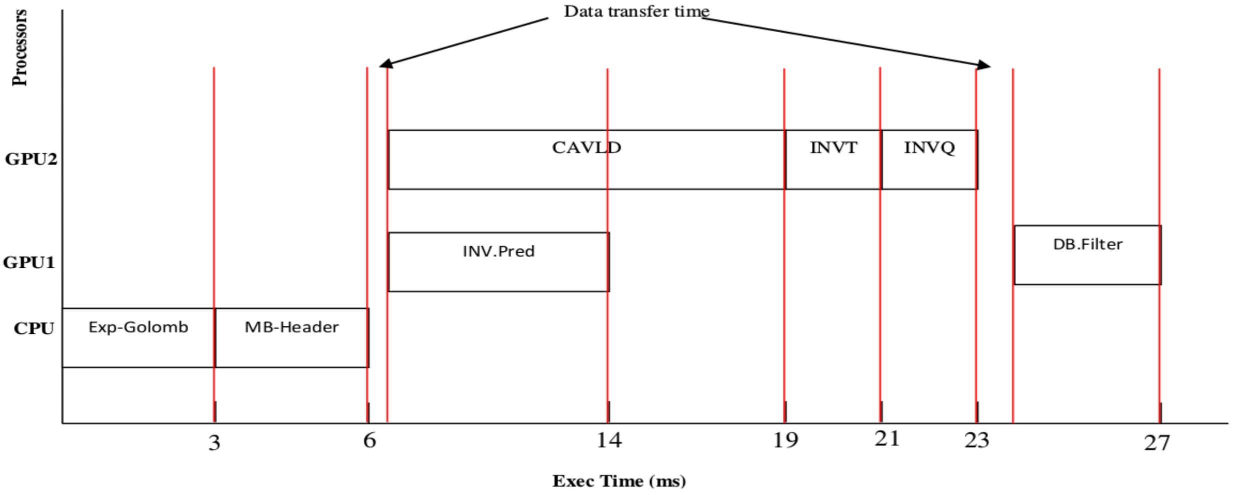


Figure 11: Execution time of the H264 decoder implementation on heterogeneous architecture HAS (multiCPU/multiGPU)

6.2 Results

Execution times with different processors CPU/GPU using HD resolutions are illustrated in figure 11.

The execution time for each task of the H264 video decoder on each type of processor taking into account the different parallelism (task parallelism, data parallelism) is shown in figure 12. This is the first work that deals with the implementation of H264 on heterogeneous architecture HAS (multiCPU/multiGPU) taking into account the different parallelism (task parallelism, data parallelism) and data dependence.

Comparing our result with previous work [13] and [16], our approach gives a better performance in terms of execution time for each H264 decoder task and for entire application.

7 Conclusion

This paper illustrated the design of the H.264 encoder on a heterogeneous architecture HAS (multiCPU/multiGPU). A high-level modeling approach based on the standard MARTE profile has been adopted. The obtained model has been analyzed by considering different parallelism, data dependence and characteristics processors (CPU/GPU). Our approach gives a better performance in terms of execution.

Finally, in the future, we are planning to integrate the idea of automatically mapping and scheduling multimedia applications such as H264 onto heterogeneous architectures taking into account different parallelism (task parallelism, data parallelism) and data dependence and the cost of data transfer.

References

[1] C. Nvidia. Compute unified device architecture programming guide.

[2] K. Corporation. The OpenCL Language. www.khronos.org/opencl, 2011.

[3] AISO/IEC. International standard. Part 10: Advanced video coding,

[4] JCT-VC. High efficiency video coding (HEVC) text specification draft 8. 10th Meeting: Stockholm, SE, 1120 July2012.

[5] C. S. Kannangara and I. E. G. Richardson and M. Bystrom and J. Solera and Y. Zhao and A. Maclennan Complexity reduction of H.264 using Lagrange Optimization Methods. IEE VIE 2005, Glasgow, UK, 2005.

[6] A. Gurhanli and S. Hung. Coarse grain parallelization of h.264 video decoder and memory bottleneck in multi-core architectures. International Journal of Computer Theory and Engineering vol. 3, no. 3, pages 375–381, 2011.

[7] K. Nishihara, A. Hatabu, and T. Moriyoshi. Parallelization of h.264 video decoder for embedded multicore processor. ICME, pages 329–332, 2008.

[8] A. Azevedo, C. Meenderinck, B. Juurlink, A. Terechko, J. Hooger-brugge, M. Alvarez, and A. Ramirez. Parallel h.264 decoding on an embedded multicore processor. HiPEAC, pages 404–418, 2009.

[9] J. Chong, N. Satish, B. Catanzaro, K. Ravindran, and K. Keutzer. Efficient parallelization of h.264 decoding with macro block level scheduling. ICME, pages 1874–1877, 2007.

[10] E. Van Der Tol, E. Jaspers, and R. Gelderblom. Mapping of h.264 decoding on a multiprocessor architecture. Image and Video Communications and Processing, pages 707–718, 2003.

[11] M. Horowitz, A. Joch, F. Kossentini, and A. Hallapuro. H.264/avc baseline profile decoder complexity analysis. IEEE Trans. Circuits Syst. Video Techn., 13(7):704–716, 2003.

[12] K. Sihn, H. Baik, J. Kim, S. Bae, and H. Song. Novel approaches to parallel h.264 decoder on symmetric multicore systems. Proceedings of the 2009 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 09, pages 2017–2020, Washington, DC, USA, 2009. IEEE Computer Society.

[13] Elias Baaklini and all, H.264 Parallel Optimization on Graphics Processors, MMEDIA 2013 : The Fifth International Conferences on Advances in Multimedia

[14] OMG, “MARTE Web Site,” 2009, www.omgmarte.org.

[15] R. Ubal, B. Jang, P. Mistry, D. Schaa, and D. Kaeli. Multi2Sim: A Simulation Framework for CPU-GPU Computing. Proc. of the 21st International Conference on Parallel Architectures and Compilation Techniques, Sep., 2012.

[16] R. Bonamy Modélisation, exploration et estimation de la consummation pour les architectures hetéerogénes reconfigurables dynamiquement HAL Id: tel-00931849 https://tel.archives-ouvertes.fr/tel-00931849v2Submitted on 17 May 2014.

[17] FFmpeg project. http://www.ffmpeg.org/.

[18] K. Suhring. H.264 reference software. http://bs.hhi.de/ suehring/tml/.

[19] TarunIyer (30 April 2013). "AMD Unveils its Heterogeneous Uniform Memory Access (hUMA) Technology". Tom's Hardware.

[20] George Kyriazis (30 August 2012). Heterogeneous System Architecture: A Technical Review (PDF) (Report). AMD.

[21] R. Ubal, B. Jang, P. Mistry, D. Schaa, and D. Kaeli. Multi2Sim: A Simulation Framework for CPU-GPU Computing. Proc. of the 21st International Conference on Parallel Architectures and Compilation Techniques, Sep., 2012.

[22] Wiegand (Ed.), T, " Draft ITU-T Recommendation H.264/AVC and Draft ISO/IEC 14496-10 AVC", Joint Video Team of ISO/IEC JTC1/SC29/WG11 &ITU-T SG16/Q.6 Doc. JVT-G050, Mar.

[23] Richardson, I.E.G.: Video Codec Design: Developing Image and Video Compression Systems. John Wiley and Sons (2002).

[24] Flierl, M., Girod, B.: Generalized B Pictures and the Draft H. 264/AVC Video-Compression Standard. IEEE Transactions on Circuits and Systems for Video Technology 13(7), 587–597(2003).

[25] Soon-kak Kwon, A. Tamhankar, K.R. Rao, "Overview of H.264/AVC / MPEG-4 Part 10", Journal of Visual Communication and Image Representation, Vol. 17, No 2 , pp 186–216, Apr. 2006.

[26] Kessentini A., Kaaniche B., Werda I., Samet A., Masmoudi N., "Low complexity intra 16x16 prediction for H.264/AVC" International Conference on Embedded Systems & Critical Applications ICESCA,2008, Tunisia.

[27] Werda I., Chaouch H, Samet A, Ben Ayed M-A, Masmoudi N., "Optimal DSP Based Integer Motion Estimation Implementation for H.264/AVC Baseline Encoder", The International Arab Journal of Information Technology, Vol. 7, No. 1, January 2010.

[28] Damak T., Werda I., Masmoudi N., Bilavarn S., "Fast prototyping H.264 deblocking filter using ESL Tools", 2011 8th International Multi-Conference on Systems, Signals &Devices,SSD.

[29] AMD Opteron Processor Family. http://www.amd.com/.

[30] AMD Evergreen Family Instruction Set Arch. (vl.Od).

[31] http://developer.amd.com/sdks/amdappsdk/documentation/.

[32] Intel Core Processor Family. http://www.intel.com.

A Parallel Adaptive Differential Evolution Algorithm for Electric Motor Design

M. Essaid[1], L. Idoumghar[1], J. Lepagnot[1], M. Brévilliers[1] and Daniel Foderean[2]

[1] IRIMAS, Université de Haute Alsace, 68093, Mulhouse, France
{mokhtar.essaid, lhassane.idoumghar, julien.lepagnot, mathieu.brevilliers}@uha.fr
[2] Department of Electerical Machines and Drives, Technical University of Cluj Napoca, Romania
daniel.fodorean@emd.utcluj.ro

Keywords: *Differential evolution, parameter adaptation, electric motors, GPU.*

1 Introduction

Differential Evolution (DE) is a powerful stochastic optimization algorithm [6]. It has been successfully applied on a wide range of optimization problems [4, 5, 11]. The iterative search process of DE relies on three search operators: mutation, crossover and selection which are controlled by three parameters (the scaling factor F, the crossover rate CR and the population size NP). Several works have studied the impact of DE parameters on its performance [6, 1–3], which revealed that their appropriate tuning would achieve a resilient performance. In this paper, we propose an on-line parameter tuning of DE along with a graphics processing unit (GPU)-based parallel implementation to reduce the computational time. The proposition has been applied to optimize the structure of a recent electric motor and compared with state-of-the-art adaptive and self-adaptive versions of DE.

2 Electric motor design optimization

The problem at hand is a recent engineering minimization problem with the aim of improving the autonomy of electric vehicles. The given optimization problem has eight variables that are depicted in Table 1.

Table 1. The geometrical variables

Symbol	Description	Variation limits
Dis	Inner stator diameter	[50; 80] mm
hjr	Rotor yoke height	[7; 15] mm
histm	Tooth isthmus	[0.5; 2] mm
hjs	Stator yoke height	[8; 15] mm
wt	Tooth width	[3.5; 8] mm
gap0	Air-gap length	[0.5; 1.5] mm
hmp	PM height	[4; 8] mm
Lm	Machines length	[100; 160] mm

The objective function to be optimized is as follows:

$$minJ(x) = -P_{out}/m_{atot} + penality \tag{1}$$

where P_{out} is the power density distribution, m_{atot} is the weight of the electric motor and

$$penality = 10^4 \sum_{i=1}^{7} C_i \tag{2}$$

In (2), C_i=0 if the constraint i is satisfied, 1 otherwise. The set of constraints is presented in Table 2.

Table 2. The problem constraints

Parameter	Symbol	Unity	Variation limits
Output power	P_{out}	W	[19995; 20005]
Current consumption	I_s	A	[20 ; 56]
Motor torque	T_m	Nm	[8.5 ; 8.6]
Motors efficiency	n	-	[0.9; 0.99]
Motors power factor	PF	-	[0.81; 0.99]
Rotor inner diameter	Dir	mm	[22; 70]
Slot filling factor	T	-	[0.1; 0.5]

3 PADE

This work proposes a GPU-based parallel adaptive version of DE called PADE. This algorithm incorporates a pool of discrete possible combinations of F and CR. Each combination has a corresponding score that represents its performance in the previous generations of the search process. Accordingly, at each generation, a roulette wheel selection is performed to select one of the best 10% combinations (combinations with the highest score) to be applied on the search operators of DE. If the combination can improve the current state of a given individual, it is rewarded. Otherwise, it is penalized. Moreover, to control NP, a linear size reduction of the population is introduced to eliminate fractions of the worst individuals over the generations. In the mutation phase, we use the well-known DE/current-to-pbest/1 proposed in [10], which has been successfully applied in several works[7, 8, 12]. DE/current-to-pbest/1 is presented as follows:

$$v_i^{G+1} = x_i^G + F.(x_{pbest}^G - x_i^G) + F.(x_{r1}^G - x_{r2}^G) \tag{3}$$

where x_i^G is the current individual, x_{pbest} is one of the best p individuals in the population, x_{r1}^G and x_{r2}^G are two random individuals. Table 3 reveals the advantage of our proposition over the unmodified DE [6] and several state-of-the-art adaptive DE algorithms, i.e. SADE [11] and EPSDE [9] where a budget of one million evaluations has been used.

Table 3. Comparison with state-of-the-art DE algorithms after 30 runs

	DE (DE/rand/1)	SADE	EPSDE	PADE
Best	-3.15e+03	-3.27e+03	-3.20e+03	**-3.33e+03**
Mean	-2.91e+03	-2.92e+03	-2.85e+03	**-3.07e+03**
Worst	-2.84e+03	-2.78e+03	-2.43e+03	-2.74e+03

In Table 4, PADE is compared with its sequential version on CPU, in terms of computational time and of best and mean fitness of the solutions found. To test the parallel implementation, we used an NVIDIA Quadro M620 GPU and an Intel i5-744HQ @ 2.80 GHZ computer. Table 4 shows that the parallel implementation can reduce the computational time and could slightly improve the results.

4 Conclusion

In this paper, we have presented a parallel adaptive DE algorithm to optimize the structure of an electric motor. The approach relies on a simple yet efficient learning technique to tune DE parameters. Moreover, the proposed algorithm, parallelized on GPU, shows significant acceleration compared to its sequential version. Promising results have been noticed when comparing with state-of-the-art DE propositions. In works under progress, we are applying PADE on other real-world problems where the acceleration, in terms of computational time, can be even more significant.

Table 4. Comparison with GPU-based implementation

NP		CPU	GPU
200	Best	-3.15e+03	-3.33e+03
	Mean	-2.91e+03	-3.07e+03
	Mean time	60.27 s	27.74 s
400	Best	-2.79e+03	-3.39e+03
	Mean	-2.76e+03	-2.95e+03
	Mean time	63.74 s	7.79 s
600	Best	-3.12e+03	-3.21e+03
	Mean	-2.75e+03	-2.90e+03
	Mean time	66.28 s	7.28 s
800	Best	-2.98e+03	-3.05e+03
	Mean	-2.76e+03	-2.85e+03
	Mean time	75.27 s	7.08 s

References

1. Roger Gämperle, Sibylle D Müller, and Petros Koumoutsakos. A parameter study for differential evolution. *Advances in intelligent systems, fuzzy systems, evolutionary computation*, 10(10):293–298, 2002.
2. Vicky Ling Huang, A Kai Qin, and Ponnuthurai N Suganthan. Self-adaptive differential evolution algorithm for constrained real-parameter optimization. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pages 17–24. IEEE, 2006.
3. A Kai Qin, Vicky Ling Huang, and Ponnuthurai N Suganthan. Differential evolution algorithm with strategy adaptation for global numerical optimization. *IEEE transactions on Evolutionary Computation*, 13(2):398–417, 2009.
4. Soumyadip Sengupta, Swagatam Das, Md Nasir, Athanasios V Vasilakos, and Witold Pedrycz. An evolutionary multiobjective sleep-scheduling scheme for differentiated coverage in wireless sensor networks. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(6):1093–1102, 2012.
5. Rainer Storn. Differential evolution design of an IIR-filter. In *Evolutionary Computation, 1996., Proceedings of IEEE International Conference on*, pages 268–273. IEEE, 1996.
6. Rainer Storn and Kenneth Price. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, 11(4):341–359, 1997.
7. Ryoji Tanabe and Alex Fukunaga. Success-history based parameter adaptation for differential evolution. In *Evolutionary Computation (CEC), 2013 IEEE Congress on*, pages 71–78. IEEE, 2013.
8. Ryoji Tanabe and Alex S Fukunaga. Improving the search performance of shade using linear population size reduction. In *Evolutionary Computation (CEC), 2014 IEEE Congress on*, pages 1658–1665. IEEE, 2014.
9. Yong Wang, Zixing Cai, and Qingfu Zhang. Differential evolution with composite trial vector generation strategies and control parameters. *IEEE Transactions on Evolutionary Computation*, 15(1):55–66, 2011.
10. Jingqiao Zhang and Arthur C Sanderson. JADE: adaptive differential evolution with optional external archive. *IEEE Transactions on evolutionary computation*, 13(5):945–958, 2009.
11. Shi-Zheng Zhao, Ponnuthurai Nagaratnam Suganthan, and Swagatam Das. Self-adaptive differential evolution with multi-trajectory search for large-scale optimization. *Soft Computing*, 15(11):2175–2185, 2011.
12. Shi-Zheng Zhao, Ponnuthurai Nagaratnam Suganthan, and Swagatam Das. Self-adaptive differential evolution with multi-trajectory search for large-scale optimization. *Soft Computing*, 15(11):2175–2185, 2011.

Virtual screening in electrostatic potential using an evolutionary algorithm

S. Puertas-Martín[1], J. L. Redondo[1], H. Pérez-Sánchez[2], and P. M. Ortigosa[1]

[1] University of Almería, Agrifood Campus of International Excellence, ceiA3, Almería, Spain
savinspm@ual.es, jlredondo@ual.es, ortigosa@ual.es
[2] Bioinformatics and High Performance Computing Research Group (BIO-HPC), Universidad Católica
San Antonio de Murcia (UCAM), Murcia, Spain
hperez@ucam.edu

1 Introduction

Nowadays, a lot of effort has been made in the health area, mainly in the pharmaceutical field, to discover more efficient and innovative drugs. This can be observed, among other factors, in the increase of scientific papers which propose new techniques amined to find new compounds. Broadly speaking, those techniques, called Virtual Screening (VS) methods, try to determine which chemical compounds, from an enormous dataset, can replace a given protein target or interact with it. This allows reducing experimentation time and in-vitro costs. Depending on the information available from the pharmacological targets and the compounds in the dataset, they can be classified between: (i) Structure-Based Virtual Screening (SBVS) techniques, which require detailed structural information about the compounds; and (ii) Ligand-Based Virtual Screening (LBVS), where only information about known ligands (actives and inactives, agonists and antagonists, etc) is exploited in order to predict new bioactive compounds against selected protein targets. LBVS methods will, therefore, consider all existing available information about known active and inactive compounds, and this information will be referred to as molecular descriptors. There exist a large number of molecular descriptors o potentials used to compare molecules, as for example Shape similarity, Electrostatic similarity, Atomic property fields, Aromatic potential, Desolvation potential, etc.

In this work, we propose the use of Optipharm, a new LBVS method to discover the most similar compound to a given target. The electrostatic potential will be considered as a measure of the existing similarity. Optipharm is a global hybrid evolutionary optimization algorithm, where a population of candidate solutions ('individuals') is generated and simulated to evolve (including interaction) until a certain halt condition met. This algorithm has been designed ad-hoc to solve this LBVS problem and some problem definition knowledge has been introduced in order to improve the search.

1.1 Electrostatic similarity

To calculate the similarity of the electrostatic potential [1], a function of the OpenEye Toolkit has been used, Zap [4]. The following is a brief overview of the most important mathematical expressions. Zap Toolkit uses a numerical solution of the Poisson equation to obtain the electrostatic potential [2]:

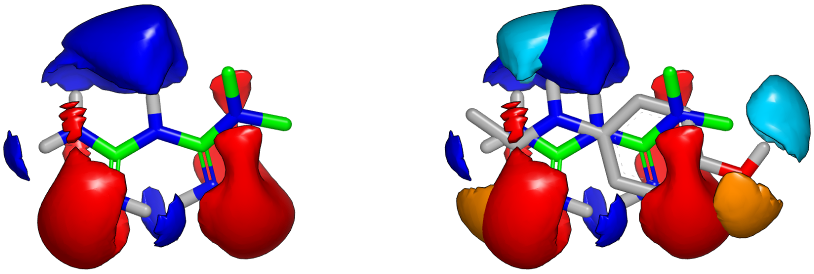
$$\nabla\{\epsilon(r)\nabla\phi(r)\} = \rho_{mol}(r) \tag{1}$$

where $\phi(r)$, $\epsilon(r)$ and $\rho_{mol}(r)$ represent the electrostatic potential, the dielectric constant and the molecular charge distribution, respectively.

Electrostatic potentials between two compounds are compared by determining the following expression:

$$E_{AB} = \int \phi^A(r)\phi^B(r)\Theta^A(r)\Theta^B(r)\boldsymbol{dr} \approx h^3 \sum_{ijk} \phi^A_{ijk}\phi^B_{ijk}\Theta^A_{ijk}\Theta^B_{ijk} \tag{2}$$

where Θ is a masking function to ensure potentials interior to the molecule are not considered as part of the comparison.



(a) Tanimoto coefficient = 1.

(b) Tanimoto coefficient = 0.78.

Fig. 1: 1a Comparison of compound DB00331 with itself and, 1b comparison of DB00331 with DB00316. In both figures, the blue and red surfaces show the positive and negative electrostatic potential of DB00331. The light blue and orange colors have been chosen to show the positive and negative electrostatic potential, respectively, of DB00316 compound, facilitating its visualization.

Finally, the final electrostatic similarity score is obtained by the standard Tanimoto coefficient:

$$Tc = \frac{V_{AB}}{V_{AA} + V_{BB} - V_{AB}} \tag{3}$$

2 Preliminary results and conclusions

The experiments have been carried out in a supercomputing cluster of the HPCA research group. Its main characteristics are 484 cores, 3900 GB of Memory and 22 TB HDD [3].

The database used for computational experiments is the Food and Drug Administration (FDA) database [6]. This database consists of approved drugs in the United States for use in humans. Such a dataset contains 1751 molecules, and some of them are used as pharmacological targets.

The preliminary results have shown the superiority of Optipharm at finding most similar compounds to a given target, as compared to the ones obtained by methods previously proposed in literature.

Figure 1 illustrates the performance of Optipharm by showing a couple of graphical results. In Subfigure 1a the electrostatic potential of compound DB00331 is computed with itself. As can be seen, the similarity score is 1. Blue and red surfaces show positive and negative potential respectively. As the score is 1, the overlap is perfect and only one compound can be seen. This is the first fire test for any LBVS, i.e. a competitive method should be able to obtain the maximum possible score (which is one), when target and compound are the same molecule. In Subfigure 1b the previous compound is represented with DB00316 obtaining a 0.78 of similarity.In this case, the light blue and orange colours have been chosen to represent the positive and negative electrostatic potential of compound DB00316 and show its differences compared to compound DB00331 (blue and red). The figures have been represented using VIDA [5], an OpenEye software.

Acknowledgements

This work was funded by grants from the Spanish Ministry of Economy and Competitiveness (TIN2015-66680-C2-1-R, CTQ2017-87974-R), Junta de Andalucía (P12-TIC301), Fundación Séneca–Agencia de Ciencia y Tecnología de la Región de Murcia under Projects 19419/PI/14 and 18946/JLI/13. Powered@NLHPC: This research was partially supported by the supercomputing infrastructure of the NLHPC (ECM-02). The authors also thankfully acknowledge the computer resources and the technical support provided by the Plataforma Andaluza de Bioinformática of the University of Málaga. This work was partially supported by the computing facilities of Extremadura Research Centre for Advanced Technologies (CETA−CIEMAT), funded by the European Regional Development Fund (ERDF). CETA−CIEMAT belongs to CIEMAT and the Government of Spain. Juana

López Redondo is a fellow of the Spanish 'Ramón y Cajal' contract program, co-financed by the European Social Fund. Savíns Puertas Martín is a fellow of the Spanish 'Formación de profesorado universitario' program, financed by the Spanish Ministry of Education, Culture and Sport.

References

1. Boström, J. and Grant, J. A. and Fjellström, O. and Thelin, A. and Gustafsson, D.: Potent Fibrinolysis Inhibitor Discovered by Shape and Electrostatic Complementarity to the Drug Tranexamic Acid. Journal of Medicinal Chemistry, 56(8), 3273–3280 (2013).
2. Böttcher, C. J. F.: Theory of Electric Polarization: Dielectrics in Static Fields. Elsevier Vol. 1, New York (1973).
3. HPCA Infraestructure. https://hpca.ual.es/en/infraestructure, (2018).
4. OpenEye Scientific Software: OEChem Toolkit. https://www.eyesopen.com, Santa Fe, Nuevo Mexico, (2018).
5. OpenEye Scientific Software: VIDA. https://www.eyesopen.com, Santa Fe, Nuevo Mexico, (2018).
6. Wishart, D. S. and Knox, C. and Guo, A. C. and Shrivastava, S. and Hassanali, M. and Stothard, P. and Chang, Z. and Woolsey, J.: DrugBank: a comprehensive resource for in silico drug discovery and exploration. Nucleic acids research, 34, D668–D672 (2006).

Improved NSGAII Based on a Multiple-Criteria Decision Analysis Method for Business Process Optimization

Nadir Mahammed[1], Sidi Mohamed Benslimane[1], Ali Ouldkradda[2],
Mahmoud Fahsi[3]

[1] (n.mahammed,s.benslimane)@esi-sba.dz *Ecole Supérieure en Informatique 08 Mai 1945, LabRI-SBA, Sidi Bel Abbes, Algeria*
[2] ould.kradda.ali@edu.univ-oran1.dz *Ahmed BenBella University of Oran 1, LRIIR laboratory, Oran, Algeria*
[3] mfahci@univ-sba.dz *Djillali Liabes University, Computer Science Department, Sidi Bel Abbes, Algeria*

Abstract. In this paper, a research was carried out on the problem of evolutionary multi objective business process optimization. It does involve (i) to construct feasible business process designs with optimum attributes, and (ii) to classify the obtained solutions using a simple and scientific approach understandable by the decision maker. The business process evolutionary multi objective optimization (BPMOO) approach involves the generation of a series of diverse optimized business process designs for the same process requirements using an evolutionary algorithm (EA). The work presented in this paper is aimed to investigate the benefits that come from the utilization of multiple-criteria decision analysis methods (MCDA) with an evolutionary multi objective optimization algorithms (EMOA) execution process. The experimental results clearly bring that the proposed optimization Framework is capable of producing an acceptable number of optimized design alternatives to simplify the decision maker's choice of solutions in a reasonable runtime.

Keywords: Multi objective optimization, evolutionary algorithm, business process, multiple-criteria decision analysis.

1 Introduction

According to [1], optimization refers to finding the best possible solution to a problem given a set of constraints. Firstly, when a single objective has to be optimized, the aim is to find the best possible solution available called "global optimum". Secondly, the case where there is not one but several objectives to optimize simultaneously. Actually, these objectives are most often in conflict with each other. These problems are called "multi-objective optimization" (MOO) that leads to a set of solutions. Therefore, a solution in a MOO is Pareto optimal [2] if it exists no other feasible solution which would decrease some criterion without causing a simultaneous increase in at least one other criterion. Such problems can be mostly solved using metaheuristics [3]. Evolutionary algorithms are particularly recommended because of their ability to handle a set of solutions simultaneously, and their capability to deal with problems of various kinds [4]. Having said that, evolutionary multi objective optimization (EMOO) was introduced in the 1980s [5], and used in a lot of disciplines, nowadays, and business process optimization (BPO) is, by no means, an exception. BPO is considered the problem of building feasible business processes (BPs) while optimizing conflicting criteria [6].

This article proposes a Framework that deals with a business process multi objective optimization dealing with 02 conflicting optimization criteria. MRS-NSGAII, for Majority Rule Sorting NSGAII is used by the Framework as enhanced EA. It tests and experiments the influence of using an MCDA in BPMOO. Section 2 presents a state of the art on BPMOO. Section 3 presents the main contribution of the paper, the optimization Framework with its proposed Fitness function and introduces MRS-NSGAII. Experiments are performed and the results are presented in Section 4. Finally, Section 5 summarizes the proposed research and provides perspectives.

2 Related Work

NSGAII is one of the most widely used evolutionary algorithms to overcome the question of multi objective (up to 03 criteria) of business processes [7]. The first work to mention is [8]. It focuses on how to appropriately allocate resources to activities in BP designs to ensure its high performance. A series of work on BPMOO with evolutionary algorithms are introduced by [9]. The proposed approach uses a formal definition of a BP. It proposed and tested a Framework using NSGAII to generate new BP designs. Thereafter, [10] to [12] present the most important work in this field. The authors have improved [9] work by adding (i) the ability to review or reconfigure any unfeasible BP design and (ii) using other EAs. They finally propose a Framework where each task can be regarded as a Web service. [13] proposed an optimization Framework using Petri networks for modeling. [14] resumed the work of [10] by modifying the mutation and crossover operators used within NSGAII. [15] are interested in a BPMOO (up to three criteria) by implementing a Framework using NSGAII with a modified crossover's operator and different selection techniques. [16] worked on a novel selection operator within NSGAII tested with a real BP scenario.

The present work is an enhancement of the optimization Framework proposed in [16]. We propose to add a step on the EA progress while generating and evaluating processes with diverse designs constructed based on a predefined business process.

3 Proposed Approach

3.1 Overall Architecture of the Proposed Framework

A BP is defined as a collective set of tasks when properly connected perform a business operation, e.g. a product or a service providing value to the organization [17]. The main elements involved are tasks, resources and attributes of the BP. The attributes provide the capability of evaluating a BP design. The problem of BPMOO can be defined as follows:

- is the BP designs search space (s BPS),
- is the Fitness function that assigns a numerical score for each BP design,
- is a set of constraints to optimize.

The aim of the optimization problem is to find either the instance of:

- Global optimal BP , such as BPS, or
- A near-optimal BP , such that δ.

Throughout the Framework course; each BP design must fulfill a certain amount of constraints. MRS-NSGAII is used to generate BP designs. Each solution set has (i) a feasible graphical representation and (ii) optimized' attributes values. Fig. 1 depicts the proposed Framework.

Create an initial population. A random population of BP designs is generated. It takes place only once in the Framework's progress. The steps 2-5 are repeated for a predefined number of iterations.

Create designs representation. For each BP design, a matrix tasks/task (Fig. 2) is generated to represent the relationship between tasks and resources composing a potential design (compared to [16] with 02 matrix).

Verify and apply the restraints. Prior to evaluate a design, the Framework verifies a set of constraints because a design might be modified thereafter. As restraints, we quote:

- A task must appear once, in each design.
- Replace or delete any task or resource useless for the design.
- Verify inputs and outputs of each BP design.

Assess designs. BP design's Fitness value is calculated based on its attributes values. The proposed Framework uses a Fitness function dealing with 02 optimization criteria. To save time, the solutions are evaluated after the restraints verification because only tasks that really participate in a BP design are taken into account.
The authors use the Fitness function proposed by [16]. It can be symbolized geometrically as the hypotenuse of the right-angled triangle formed by a_1 and a_2.

(1)

-
- : BP design optimization criterion.
- : attribute of a task BP design.

- : Number of tasks in a BP design.
- are normalized.

Perform EA. MRS-NSGAII is applied (simulated binary tournament selection, simulated binary crossover and mutation operators). The process does not check whether a solution is feasible (step 3). Subsection III-B introduces MRS-NSGAII and its inner functioning.

3.2 MRS-NSGAII

MRS-NSGAII is an enhanced version of NSGAII proposed by [17] (see Fig. 3). The authors propose to add the majority rule sorting method (MR-sort) to the non-dominated sorting stage into NSGAII execution. MR-sort is a simplified version of the ELECTRE TRI sorting model [19][20]. The general principle of MR-sort is to assign alternatives by comparing their performances to those of profiles delimiting proposed categories. To the authors' knowledge this technique has never been used with NSGAII and particularly in a BPMOO, by the past. MRS-NSGAII can be summarized as follows:

1. A parent population called is randomly generated and an offspring population is created from it.
2. Both populations and are combined into population with population size.
3. The population is categorized by going through the MR-sort model where all members are classified and put into categories.
4. The non-dominated sorting is applied on all categories except the last category (i.e. the worse according to the sorting by MR-sort).
5. The best remaining N individuals from are selected using the crowding distance and so form the next generation's parent population .
6. The steps 1-5 are repeated until the stopping criteria have been satisfied.

MR-sort method works as follows [21]: Let X be a set of objects (i.e. BP designs) evaluated on n ordered –optimization- criteria, $\}$. We assume that X is the Cartesian product of the criteria scales, . An object $a \in X$ is a vector $)$, where for all j. The ordered categories which the objects are assigned to by the MR-sort method are denoted by C_h, with p. Category is delimited by its lower limit profile and its upper limit profile $b_h$which is also the lower limit profile of category (provided p). The profile b_h is the vector of criterion values , with $b_{h,j} \in X_j$ for all j. We denote by P $\}$, the list of category indices. It is assumed that the profiles dominate one another, i.e. , for $h = \{1, \dots, p\}$ and $j = \{1, \dots, n\}$.

An object is assigned to a category if its criterion values are at least as good as the category lower profile values on a weighted majority of criteria while this condition is not fulfilled when the object's criterion values are compared to the category upper profile values. In the former case, we say that the object is *preferred* to the profile, while, in the latter, it is not.

Formally, if an object $a \in X$ is *preferred* to a profile b_h, it's denoted by $a \geq b_h$. Object a is preferred to profile b_h whenever the following condition is met:

$$a \geq b_h \iff \sum_{j:a_j \geq b_{h,j}} w_j \geq \lambda \qquad (2)$$

where w_j is the non negative weight associated with criterion j, for all j and λ sets a majority level. The weights satisfy the normalization condition $\sum_{j \in F} w_j = 1$, λ is called the *majority threshold*; it satisfies $\lambda \in [1/2, 1]$.

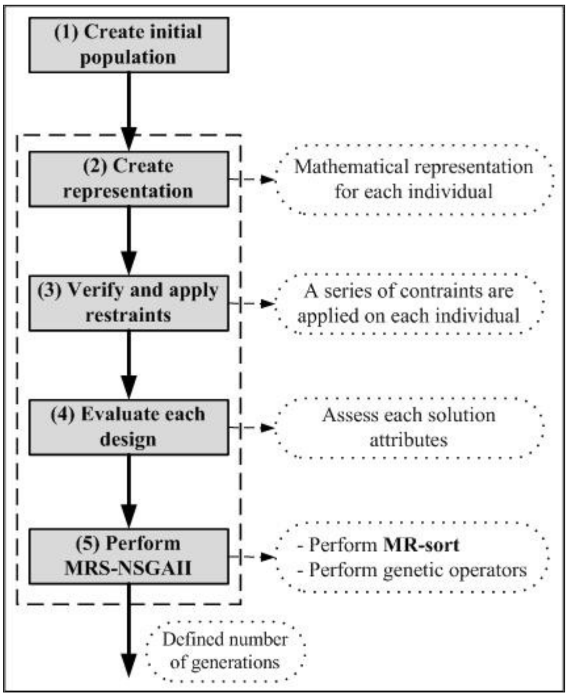
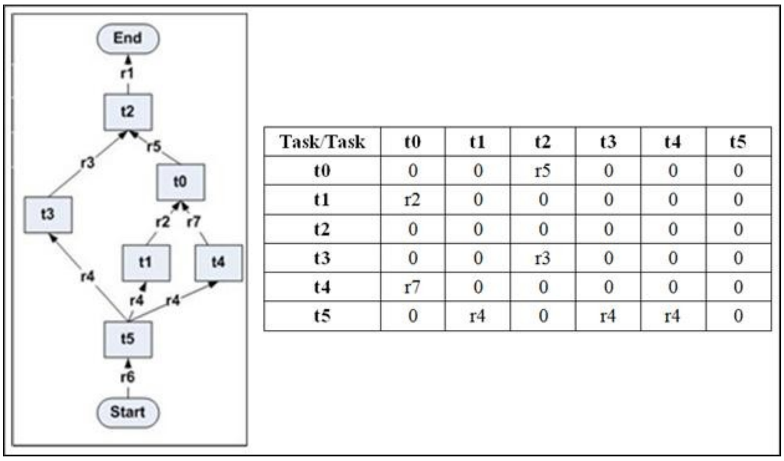

(1) Create initial population
(2) Create representation
Mathematical representation for each individual
(3) Verify and apply restraints
A series of contraints are applied on each individual
(4) Evaluate each design
Assess each solution attributes
(5) Perform MRS-NSGAII
- Perform MR-sort
- Perform genetic operators
Defined number of generations

Fig. 1. Optimization Framework main steps



Task/Task	t0	t1	t2	t3	t4	t5
t0	0	0	r5	0	0	0
t1	r2	0	0	0	0	0
t2	0	0	0	0	0	0
t3	0	0	r3	0	0	0
t4	r7	0	0	0	0	0
t5	0	r4	0	r4	r4	0

Fig. 2. Task/task matrix

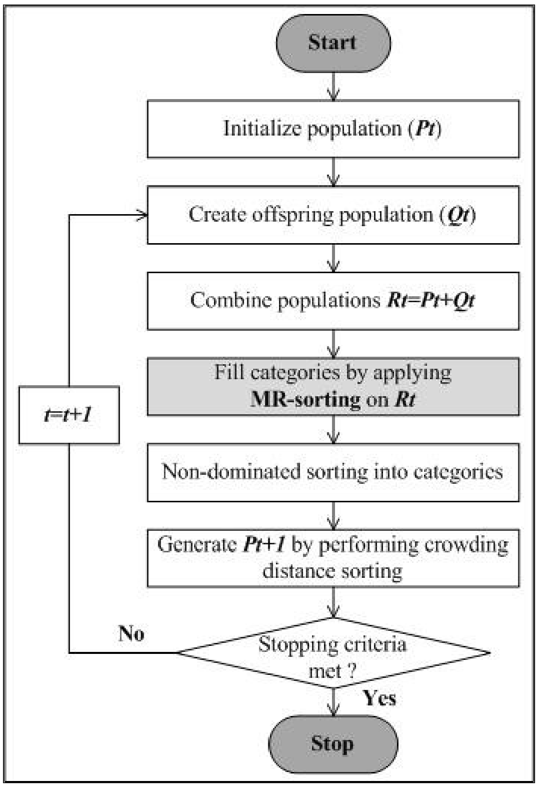

Start
Initialize population (Pt)
Create offspring population (Qt)
Combine populations Rt=Pt+Qt
Fill categories by applying MR-sorting on Rt
Non-dominated sorting into categories
Generate Pt+1 by performing crowding distance sorting
Stopping criteria met ?
No
Yes
t=t+1
Stop

Fig. 3. MRS-NSGAII main steps

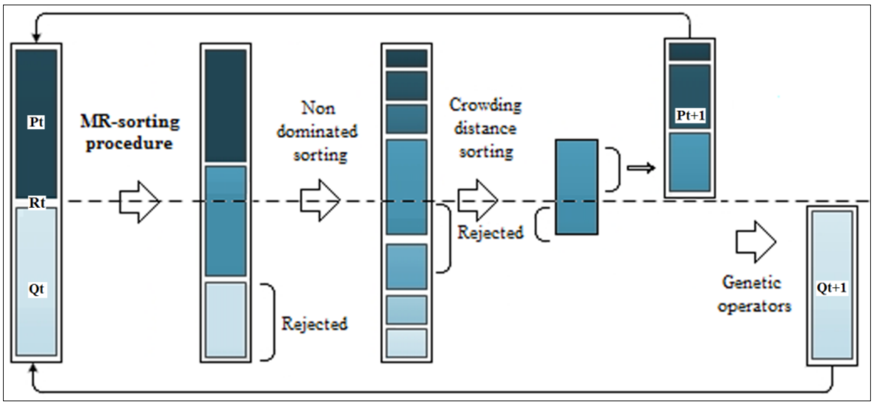


Fig. 4. MRS-NSGAII illustration

4 Experimentation and Results

In order to generate satisfactory results, the proposed Framework needs to achieve two goals (i) obtain optimal business process designs by converging to the Pareto-optimal front and (ii) obtain a variety of different sizes of BP designs while maintaining the population diversity. This suggests that the features of the problem that require more attentions are:

- The number of feasible non-dominated solutions.
- The different acceptable BP designs sizes.
- The execution time.

Each of these problem features is related with the performance goals of the Framework. The optimization Framework is expected to increase the quality of generated solutions in shorter time periods. The work presented is aimed to investigate the benefits that come from the utilization of MRS-NSGAII. This feature puts to test both the convergence and diversity capabilities of MRS-NSGAII execution. It must not content itself by discovering feasible solutions but also to converge towards the optimal, in reasonable time frames.

Table 1 shows the parameters used by the proposed Framework. The problem is set up to deal with 02 criteria a_1 (x-axis) and a_2 (y-axis). MRS-NSGAII performs 500 iterations, it might seem excessively low but initial experiments showed that it produced better quality results in comparison with higher numbers and in a timely fashion. Initial population is limited to 500. Table 2 shows the parameters of the test scenarios. To apply correctly MR-sort, we propose 03 categories: "*Good*", "*medium*" and "*bad*" ranked from most important (to the decision maker) to least important, respectively. The limit profiles are resulting from [16] experiments.

NetBeans 8.1 IDE and Java 8 on a MSI GT70 laptop have been used to perform the experimentation. This article proposes to add MR-sort to the canonical non-dominated sorting used within NSGAII, and compare the results using traditional NSGAII.

Table 1. Framework parameters

Parameter	**Value**
Evolutionary algorithm	MRS-NSGAII
Simulated Binary crossover (probability)	0.8
One point mutation (probability)	0.2
Simulated binary tournament selection	-
Optimization criteria	$\{a_1, \quad \}$
Categories	$\{C$: good, : medium, : bad$\}$
Limit profiles Scenario A	$\{b$ 1003, =1801, 4935$\}$
Limit profiles Scenario B	$\{b$ =1663, 0, 5191$\}$
Limit profiles Scenario C	$\{b$ 1788, 2001, 5267$\}$
Limit profiles Scenario D	$\{b$ 1850, 2113, 5374$\}$
	0.86

Table 2. Scenarios test parameters

Parameter	Scenario A	Scenario B	Scenario C	Scenario D
Library	30	100	500	1000
Resource $(r$	9	30	30	30
Objective	2	2	2	2
	[100 200]	[100 200]	[100 200]	[100 200]
	[300 400]	[300 400]	[300 400]	[300 400]

In summary, MR-sort has proved to enhance the performance of MRS-NSGAII used by the proposed Framework in the optimization of the business processes scenarios by reducing its execution time while finding more non dominated solutions.

In more detail (Table 4):

- MR-sort has a very good impact on the runtime of the Framework for all scenarios. Resulted in execution time decrease rate to 44,44% with scenario B.
- MR-sort method assists NSGAII in generating more non-dominated solutions. The increase rate of the solutions comes up to 56.41% for scenario C.
- The Framework has better solutions using MRS-NSGAII both from the point of you of the convergence towards optimal solutions and maintaining the population diversity.

Table 3. Optimization data for different scenario

	EA	Average time (millisecond)	Generated non-dominated solutions
Scenario A	NSGAII	4927	25
	MRS-NSGAII	3333	36
Scenario B	NSGAII	5247	37
	MRS-NSGAII	2915	49
Scenario C	NSGAII	5355	39
	MRS-NSGAII	3679	61
Scenario D	NSGAII	5547	84
	MRS-NSGAII	3954	89

Table 4. MSR-NSGAII optimization performance

Scenario A	Time decrease (%)	32,35
	Number of solutions increase (%)	36
Scenario B	Time decrease (%)	44,44
	Number of solutions increase (%)	32,43
Scenario C	Time decrease (%)	31,29
	Number of solutions increase (%)	56,41
Scenario D	Time decrease (%)	28,71
	Number of solutions increase (%)	5,95

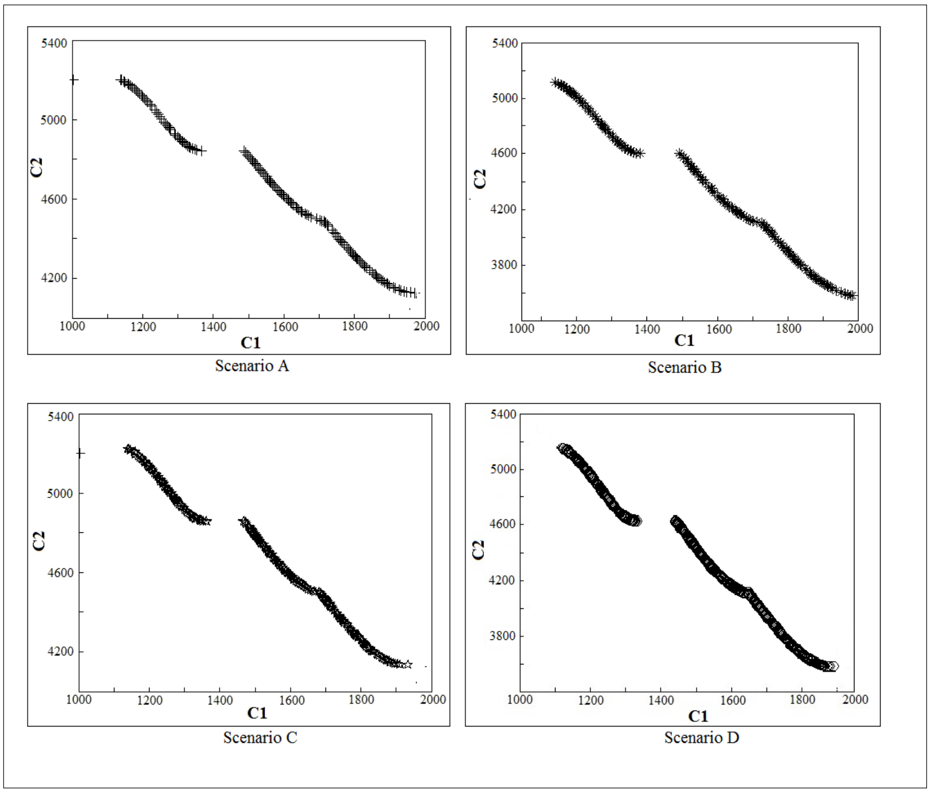


Fig. 5. All scenarios results

5 Conclusion

Knowing that the optimization of business processes is as important as delicate to deal with for modern organizations, this article presents an enhancement of a Framework capable of generating optimal feasible BP designs. It involves a quantitative representation for each solution, a modified optimization EA that generates diverse optimized designs using a MCDA method for the categorization of the solutions. The results have demonstrated that the Framework with the aid of the MR-sort method has increased its capability of generating diverse solutions and selecting the optimal ones in less time. Which pave the way to further experimentations as using another type of MCDA (e.g. for ranking), adding more optimization criteria and rewrite the Fitness function as well. Using MR-sort method needs to set several parameters and it is not easy for a decision maker to assess such parameters. In the present article, we choose to assess these parameters regarding to previous work and automate the proceeding is the next step to achieve.

6 References

1. Coello, C. A. C. (2006). Evolutionary Multi-Objective Optimization: A Historical View of the Field. *IEEE Computational Intelligence Magazine*, 1(1), 28-36.
2. Pareto V. (1896). Cours d'Economie Politique, volume I and II. F. Rouge, Lausanne.
3. Coello, C. A. C., Lamont, G. B., & Van Veldhuizen, D. A. (2007). Evolutionary Algorithms for Solving Multi-Objective Problems (Vol. 5). New York: Springer.
4. Deb, K., & Goel, T. (2001). Controlled Elitist Non-Dominated Sorting Genetic Algorithms for Better Convergence. In *EMOO* (pp. 67-81). Springer Berlin/Heidelberg.
5. Schaffer, J. D. (1985). Multiple Objective Optimization with Vector Evaluated Genetic Algorithm. *In 1st International Conference of GA and their Application* (pp. 93-100).
6. Mahammed, N., & Benslimane, S. M. (2016). Toward Multi Criteria Optimization of Business Processes Design. In *MEDI'2016* (pp. 98-107). Springer Inter. Publishing.
7. Goel, T., Vaidyanathan, R., Haftka, R. T., Shyy, W., Queipo, N. V., & Tucker, K. (2007). Response Surface Approximation of Pareto Optimal Front in Multi-Objective Optimization. *Computer Methods in Applied Mechanics and Engineering*, 196(4), 879-893.
8. Hofacker, I., & Vetschera, R. (2001), Algorithmical Approaches to Business Process Design. *Computers & Operations Research*, Vol. 28(13), pp. 1253-1275.
9. Vergidis, K., Tiwari, A., & Majeed, B. (2006), Business Process Improvement Using Multi-Objective Optimisation. *BT Technology Journal*, Vol. *24*(2), pp. 229-235.
10. Vergidis, K., Saxena, D., & Tiwari, A. (2012), An Evolutionary Multi-Objective Framework For Business Process Optimisation. *Applied Soft Comp*., Vol. 12(8), pp. 2638-2653.
11. Vergidis, K., Turner, C., Alechnovic, A., & Tiwari, A. (2015), An Automated Optimisation Framework For The Development Of Re-Configurable Business Processes: A Web Services Approach. *Int J Comp Integ M*, Vol. 28(1), pp. 41-58.
12. Georgoulakos, K., Vergidis, K., Tsakalidis, G., & Samaras, N. (2017). Evolutionary Multi-Objective Optimization of Business Process Designs with Pre-Processing. In *CEC'*2017 IEEE Congress on (pp. 897-904). IEEE.
13. Wibig, M. (2012), Dynamic Programming and Genetic Algorithm for Business Processes Optimisation, Intern. *Journal of Intelligent Systems and Applications*, Vol. 5(1), pp. 44.
14. Farsani, S.T., Aboutalebi, M., Motameni, H. (2013), Customizing NSGAII to Optimize Business Processes Designs. *Research Journal of Recent Sciences*, Vol.2, pp. 74-79.
15. Mahammed, N., & Benslimane, S. M. (2017). An Evolutionary Algorithm Based Approach for Business Process Multi-Criteria Optimization. *IJOCI*, 7(2), 34-53.
16. Mahammed, N. & Benslimane, S.M. (2018). Evolutionary Multi-Objective Optimization of Business Process Designs with MA-NSGAII. In *CIIA*'2018. Springer-Verlag IFIP AICT (in press).
17. Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T.A.M.T. (2002). A Fast and Elitist Multi Objective Genetic Algorithm: NSGA-II. *IEEE trans. on Evol. Comp*., 6(2), 182-197.
18. Bouyssou, D., & Marchant, T. (2007). An axiomatic approach to noncompensatory sorting methods in MCDM, I. *European Journal of Operational Research*, *178*(1), 217-245.
19. Yu, W. (1992). *Aide multicritère à la décision dans le cadre de la problématique du tri: concepts, méthodes et applications* (Doctoral dissertation, Paris 9).
20. Roy, B., & Bouyssou, D. Aide multicritère à la décision: méthodes et cas, 1993. *Economica, Paris*.
21. Sobrie, O., Mousseau, V., & Pirlot, M. (2015, September). Learning the parameters of a non compensatory sorting model. In *ADT'*2015 (pp. 153-170). Springer, Cham.

Efficient Generic Support for Global Routing Constraints in Constraint-Based Local Search Frameworks

Quentin Meurisse[1] Renaud De Landtsheer[2] and Christophe Ponsard[2]

[1] Faculty of Science, University of Mons, Mons, Belgium
quentin.meurisse@student.umons.ac.be
[2] CETIC Research Centre, Gosselies, Belgium
{renaud.delandtsheer,christophe.ponsard}@cetic.be

Abstract. Global constraint are very popular mechanism in local search because they achieve very good complexity, notably for routing problems. Our concern is on building generic re-usable framework for local search and builds on the concept of Constraint-Based Local Search. We showed how to embed global constraints for routing into CBLS framework, thanks to an extension of this framework, namely a variable of type "sequence of integers". Developing a global constraint for our framework still requires two things. First designing a differentiation algorithm, then embedding this algorithm into the targeted framework. In this paper we identify a generic stereotype of differentiation used by global constraints and propose a generic support for this stereotype. The stereotype is an algebraic group. Our generic framework is basically an abstract class with the place holders for this algebraic group. Once this class is properly extended, it gives rise to a fully usable and efficient global constraint featuring some form differentiation. Our framework is illustrated on classical examples as well as on an intricate global constraint for vehicle capacity.

1 Introduction

Efficiently evaluating constraint in the context of vehicle routing optimization by local search relies on so-called *global constraint* algorithms [1–3]. A classic example of global constraint is one that maintains the length of the route in routing optimization through an incremental computation [1]. For example, when considering the flip of a portion of route (i.e. a, b, c, d becomes d, c, b, a or $a, b, c, d, e, f, g, h, i$ becomes $a, b, c, \mathbf{g}, \mathbf{f}, \mathbf{e}, \mathbf{d}, h, i$) and considering a symmetric distance, a smart global constraint is able to update the route length in $O(1)$-time because the length of the flipped segment is the same in both directions.

Some global constraints can also rely on *differentiation.* At some point they are informed about a *current solution* and that lots of neighbour solution will be explored around this current solution. They typically perform some pre-computation and use it to quickly evaluate the explored neighbour solutions. In the case of the asymmetric distance matrix, the pre-computation typically associates the forward and the backward distance to each node. The forward (resp. backward) distance associated to a node represents the distance travelled by the vehicle from the start point to the node (resp. the distance travelled from the node to the start point if the whole route was flipped). When a subsequence is flipped, the old (resp. new) length of the flipped segment is the difference between the forward (res. backward) distances associated to the nodes at both ends of the flipped segment. These two distances can be computed in $O(1)$ -time, thanks to the pre-computation.

Such efficient algorithms are most of the time implemented into custom-made solvers, usually for run time efficiency reasons. This raises two concerns; first: their implementation is often not reusable, and second: they are costly to implement since these algorithms can be complex.

Our research line is to provide a modular framework enabling to quickly and easily develop, benchmark and evolve search procedures by assembling algorithmically efficient more generic building bricks. We developed the OscaR.cbls featuring a constraint-based modelling language that notably features variables of type "sequence of integers" (*SeqVar*) [4–6]. This specific variable type enables us to embed efficient global constraints into our framework because these global constraint receive symbolic, incremental information about the neighbours that are explored by the search procedure. This framework also features some checkpoint mechanism that the search procedure uses to tell global constraint when they can perform pre-computations, so that they can use differentiation approaches.

Our framework also features a domain-specific language for assembling complex search procedures as combination of neighbourhoods [7]. It notably involves a cross-product operator between neighbourhoods [8]. The consequence of this cross-product is that checkpoints might actually be stacked. Standard neighbourhoods are also provided, including routing neighbourhoods [9] and scheduling algorithms [10].

This paper builds on top of this framework and proposes a generic stereotype for implementing global constraints on top of our SeqVar by defining an algebraic group. It provides support for triggering and managing pre-computations so that the only concern to address when implementing a new global constraint is exclusively related to the specific aspects of the global constraints.

For the concrete illustration of our work, we will use our own OscaR.CBLS framework. However the design presented here can be transposed to other CBLS frameworks provided they support the above stated foundations related to the sequence variable.

The structure of this paper is as follows. Section 2 provides some minimal background over OscaR.CBLS and sequence variable. Based on this, Section 3 will present our generic support for global constraints. Section 4 will illustrate our framework on a more elaborated global constraint, namely the capacity constraints. Finally Section 5 will draw some conclusions and discuss our future work.

2 Background

2.1 Constraint-Based Local Search

Like other local search approaches, CBLS relies on a *model* and a *search procedure*. CBLS frameworks may offer support for both of these two aspects.

In CBLS frameworks, the *model* is composed of *variables* (integers and set of integers at this point), and *invariants*, which are directed constraints maintaining one or more output variables according to the specification they implement and according to the value of one or more input variables. A classical invariant is *Sum*. It has an array of integer variables as input, and a single integer variable as output. The value of the output variable is maintained by the invariant to be the sum of the values of all the input variables. This means that the invariant sets the value of the output variable, and adjust it according to the changes of the input variables. This is generally implemented through incremental algorithms. For instance, when the *Sum* invariant defined here is notified that one of its input variables has changed its value, it computes the delta on this variable between the new value and the old value, and updates the value of its output variable by incrementing it by this delta. OscaR.cbls for instance has a library of roughly 80 invariants.

The model is declared at start-up by some application-specific code that instantiates the variables and invariants. The input and output variables of each invariant are also specified at this stage.

During the search, the model is active: if the value of some input variable changes, this change is propagated to the other variables of the model through a process called *propagation* that is managed by the underlying CBLS framework. Propagation is performed in such a way that a variable is updated at most once, and only if it needs to be updated.

2.2 Sequence Variable

The OscaR.CBLS framework supports a variable of type “sequence of integers” [6]. This section gives a quick summary of the sequence variable to make the paper self contained. The proposed variable comes with a tailored data-structure enabling complex moves such as flips to be performed on the value of a single variable and described in a symbolic way as an update performed on the previous value of the variable. Global constraints are then aware of changes performed to these variable in a symbolic form. In order to enable very efficient implementation of global constraints, potentially having $O(1)$-time complexity for evaluating neighbour solutions, the data structure used for representing sequences exhibits similar complexity. Sequence values are represented by non-mutable data structures that are specifically developed for the context of local search.

The sequence variable replicates the three incremental operations supported by the sequence value (insert, remove, move) in a mutable form, and they also can be assigned a sequence value.

The sequence variable also features a complete checkpoint mechanism that serves to signal when pre-computations should be performed and also makes it possible to use a *Rollback* operation that restores the variable to the value it had at the checkpoint.

Upon propagation, sequence variables notify their value updates to the global constraint listening to them. They communicate in the form of a *SeqUpdate*. The object model of SeqUpdates is shown in Figure 1.

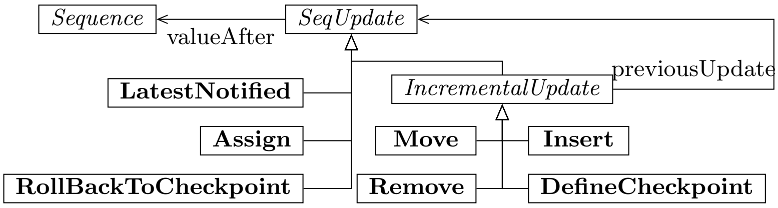


Fig. 1: Class diagram of the notifications used by sequence variables

2.3 VRP Encoding Using a SeqVar

To represent our VRP, we will take the following assumptions and conventions:
- there are v vehicles, numbered for 0 to $v1$
- there are n nodes, numbered for 0 to $n1$
- we assume $v < n$
- notation abuse: $n - 1$ and $n + 1$ will be interpreted as the next node in the route (and not as an integer addition)

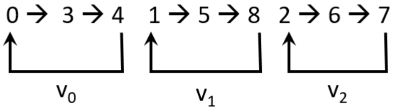


Fig. 2: Encoding of VRP using SeqVar

A unique SeqVar is used to represent the whole problem as shown in Figure 2. It is composed of the concatenation of all vehicle routes: "road of vehicle 0" → "road of vehicle 1" → ... → "road of vehicle $v - 1$" as depicted in Figure 2. For a given vehicle w, we also define the starting point to be w (same number) and will refer to the endpoint as $w*$ (with an implicit return to start point).

3 Generic Global Constraints Mechanism

3.1 Identifying a Recurring Stereotype of Global Constraints

We noticed that lots of global constraints on sequences rely on pre-computation and actually define a mathematical structure (T,+,−,flip) where T is a type, + and − are opposite of one another. In mathematical terms, this structure is a non-Abelian group with neutral element, inverse values (for each x there exists a $-x$). A group means + is associative, however we do not assume that it is commutative, thats why it is non-Abelian. For conciseness, we assume that +, − and flip are $O(1)$ -time operations.

Each node in the sequence has an associated value of type T, represented by the function:

value(node) : T

The classical pre-computation performed by global invariant is to associate to each node a value pre-computed(node) defined as follows:

$\text{precomputed(node)} : T = \sum_{n \in \text{PathBetween(vehicleStart,node)}} \text{value}(n)$

These pre-computed values can be generated in a single pass over the current route, this takes $O(n)$ -time by iterating over the sequence.

Flipping a segment is a common operation and can also be defined as an extra operation when it can managed independently of the + and − operation:

$\text{flip} : T \to T$

When considering a fraction of path represented by a pair of nodes (a, b), we define its value as the sum of the value of all node that constitute it:

$$\text{segmentValue}(a, b) =_{def} \sum_{n \in \text{PathBetween(a,b)}} value(n)$$

We can reason on the definition of precomputed to extract in $O(1)$ -time the sum of all value of nodes between a and b:

$$\begin{aligned} \text{segmentValue}(a, b) &= \text{precomputed}(b) - \text{precomputed}(a) + \text{value}(a) \\ &\quad \textit{//if } (a, b) \textit{ was not flipped since pre- computation was generated} \\ &= \text{flip}(\text{precomputed}(b) - \text{precomputed}(a)) + \text{value}(a) \\ &\quad \textit{//if } (a, b) \textit{ was flipped since pre- computation was generated} \end{aligned}$$

For any sequence of node, its cost can easily be computed according to the global constraints as the sum of the cost of each subsequence that compose it.

The global gain of this approach is that pre-computations are performed once per neighbourhood exploration and cost O(n)-time. They can be queried in $O(1)$-time for each segment of the sequence, and therefore for each explored neighbour. This is a winning approach if we explore more than $O(n)$ neighbours and if the sequence is not fragmented into too many sub-sequences according to the specific data structure presented later in this paper.

The template for defining a global constraint is therefore defined as described in Table 1.

Table 1: Template with explicit flip operator

Template	**Definition**
T	Type of computation associate to carry out global invariant computation
value : $node \to T$	Returns computation associated with a node
$+ : T \times T \to T$	Operator for combining different nodes/segments (without flip)
$- : T \times T \to T$	Operator to removing a node/segment (without flip)
flip : $T \to T$	Operator to flip a segment

3.2 Example: Routing Distance

A classic example of global constraint is the constraint that maintains the length of the route in routing optimization. It takes a distance matrix specifying the distance between each pair of nodes of the routing problem and the current route as input. Assuming the most general case with an asymmetric distance matrix, we can perform the pre-computation described in the introduction to ensure that the overall route length can be updated in $O(1)$-time in case a portion of the route is flipped. To do this, we associate a couple of integers to each node:

- the length of the incoming hop
- the length of the incoming hop supposing it is taken in reverse direction

This defines T as (Int, Int). We also define + and − as the pairwise sum and difference on the couple of integers.

Type T is defined as the constraint that maintains both the forward and backward distances at each node ($distanceFW$ and $distanceBW$). The segment boundaries are also recorded ($firstNode$ and $endNode$). Considering $m(p1, p2)$ and $m(p2, p1)$ respectively return the forward and backward distance between points $p1$ and $p2$, it is easy to express the + and − operation. The flip operation simply swap forward and backward distances and segment boundaries. Finally value is trivial to express as distances are zero and both boundaries are on the considered node.

The fully instantiated template is presented in Table 2. Note that type T has been further enriched with the first and last node of the considered route.

Table 2: Template instantiated for the routing distance global constraint

Template	**Definition**
T	$(distanceFW : Int, distanceBW : Int, firstNode : Int, lastNode : Int)$
value : $node \rightarrow T$	$(distanceFW = 0, distanceBW = 0, firstNode = node, lastNode = node)$
$+ : T \times T \rightarrow T$	$(a, b, fn1, ln1) + (c, d, fn2, ln2) \rightarrow (a + c + m(ln1, fn2), b + d + m(fn2, ln1), fn1, ln2)$
$- : T \times T \rightarrow T$	$(a, b, fn1, ln1) - (c, d, fn2, ln2)$ $\rightarrow (a - c - m(ln2, succ(ln2)), b - d - m(succ(ln2), ln2), succ(ln2), ln1)$
flip : $T \rightarrow T$	$(a, b, fn1, ln1) \rightarrow (b, a, ln1, fn1)$

3.3 Re-using Pre-computations

The global strategy is to ensure that pre-computation is done before exploring a neighbourhood. When a neighbour is evaluated, the relevant subsequence is identified and the differential formula using pre-computation is applied on it. When dealing with simple neighbourhoods (like 2-opt,3-opt), it is trivial to identify the involved subsequence. However this becomes more tricky when the exploration is more complex and relies on cross-product of neighbourhoods as described in Figure 3. In this case, we do not want to perform pre-computation at each intermediary level.

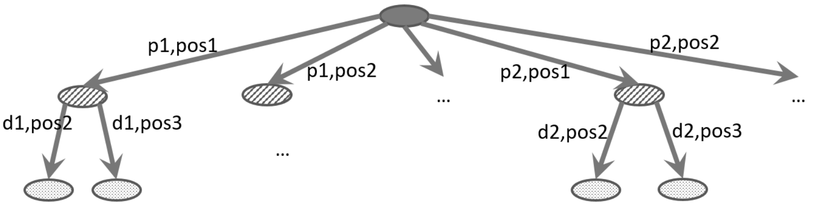


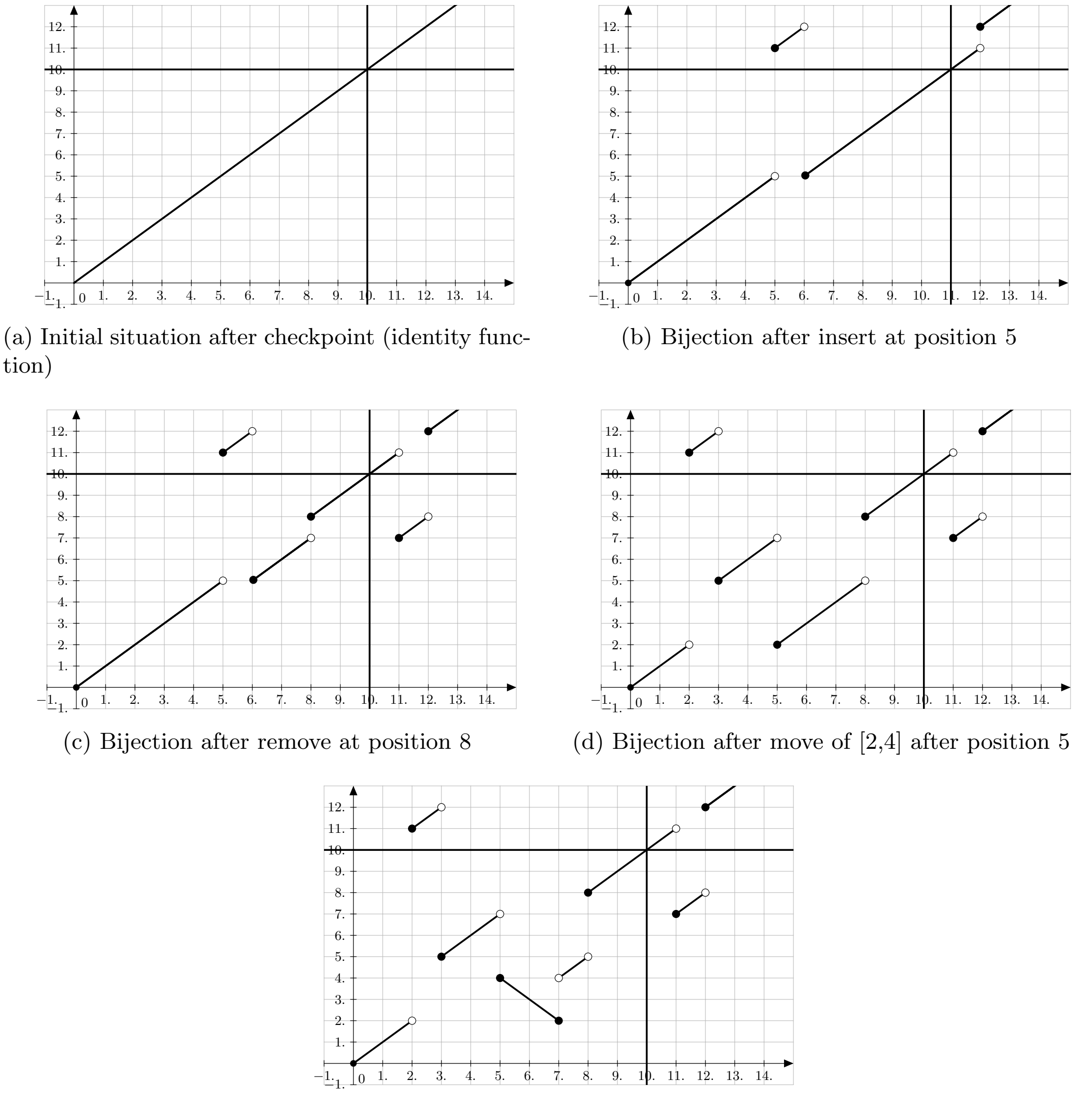
Fig. 3: Typical search tree in the presence of cross-products

In order to re-use pre-computation when updating the sequence or when exploring search trees, we maintain a piecewise affine bijection that maps the node position in the current sequence to the node position at the checkpoint level 0. At this level, the bijection is initialized to the identity function. As the sequence is being updated, this bijection will progressively become more and more fragmented as illustrated in Figure 4. The position of the last routed node at the checkpoint level 0 is kept in memory and is named ω. It is represented in the figures by the vertical line.

In practice, we only care about positions between 0 and ω given only those contribute to define the current violation. Since the image of inserted nodes is always strictly greater than α, it is easy to distinguish nodes with pre-calculations from those without pre-calculations. Segment present in the bijection can be classified in three categories: (1) segment having pre-calculations (2) segment having pre-calculations and a negative slope (i.e. flipped) and (3) segments without pre-calculations.

Assuming the group $(T, +, -, \text{flip})$ has been defined, the value can be computed through combining contribution from each segment as:

$$\sum_{\substack{[xy]\text{, segment of bijection} \\ \text{on domain defined for vehicle}}} \begin{matrix}\text{(pre-computations for y)} - \text{(pre-computations for x)} \\ \text{flipped if segment } [xy] \text{ has a negative slope}\end{matrix}$$



(a) Initial situation after checkpoint (identity function)

(b) Bijection after insert at position 5

(c) Bijection after remove at position 8

(d) Bijection after move of [2,4] after position 5

(e) Bijection after flip of [5,7]

Fig. 4: Evolution of the bijection from position now to position at pre-computation time

Note that if a node was not in the sequence when the pre-computation was performed, its value must be inserted in the sum as is. To ease the update, two interesting properties are the following ones (see [11] for their proofs.)

Property 1 (Pre-computation completeness on a segment) *Given seg, a segment of the bijection whose ends have pre-calculations. All points of seg have pre-calculations.*

Property 2 (Pre-computation relation with initial value) *Let* $\mathrm{value}_n(p)$ *be the value of pre-calculation for node p when performed after n route operation (no considering flips) since checkpoint.*
Let $\mathrm{value}_0(p)$ *be the initial value of the pre-computation performed at the checkpoint.*
Let x and y be two points with pre-calculations and such that they belong to the same segment of the bijection. Then, we have:

$$\mathrm{value}_n(y) - \mathrm{value}_n(x) = \mathrm{value}_0(y) - \mathrm{value}_0(x)$$

The proposition 1 guarantees all pre-calculations have been done entirely on a segment while proposition 2 enables the reuse of the pre-calculations. Note however that segments with a negative slope must be managed separately.

4 Application - Capacity Invariant for VRP

4.1 Problem Definition

We consider a fleet composed of v vehicles and n nodes to be visited. Each node p is characterised by a weight $w(p)$ which is the number of units that needs to be taken in or out of the vehicle at this node. All units are considered equivalent and all vehicles have the same total capacity c.

The goal is to maintain the capacity violation invariant, i.e the capacity overflow with respect to the maximal capacity. Given the above convention this invariant can be expressed as follows for a given vehicle w,

$$\text{overflow(w)} = \sum_{p=w}^{w*} \max(0, content_w(p)\text{-c})$$

where content(w) is the current capacity at the output of node w

$$\text{content(w)} = \sum_{p=w}^{p} \text{w(p)}$$

Table 3 shows a simple example of route with weight resulting in some violation level at different points of the route. This table suggest a naive approach where the total violation is computed by systematic iteration over the whole route which makes it O(n).

Table 3: Example of route with capacity violation. Total violation is 6

node	0	1	2	3	4	5	6	7
w(node)	3	1	2	−4	−3	6	1	−1
content	3	4	6	2	−1	5	6	5
overflow	0	0	2	0	0	1	2	1

Figure 5 illustrates the evolution of the capacity violation graphically as a function of the route node. Actually it has a simple and nice graphical interpretation: it is the surface of the curve above an horizontal line at the maximal capacity level. The surface value is 6 as computed in Table 3.

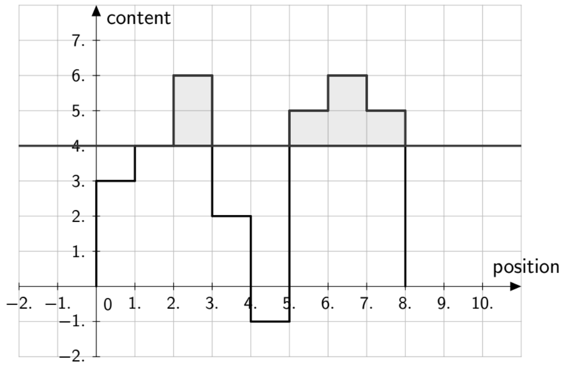


Fig. 5: Graphical representation of the capacity violation invariant

4.2 Identification of Pre-Computation and Ad-Hoc Implementation

In order to compute the capacity violation in $O(1)$, it is required that each vehicle has quick access to the full history of its content at each point of his route. Moreover this history must be efficient to compute (time efficiency) and represented in a condensed form (space efficiency).

This can actually be achieved using a Red-Black tree, i.e. a specialised form of binary search tree which is self-balancing and exhibit $O(\log \text{n})$ complexity for search, insert and delete operations. Each node m will contain such a tree to store couples of the form $(value, count)$ meaning that the considered node has had *count* times the content *value*. This data structure has the wished properties because:

- given that we have all values and their multiplicity, the constraint violation is computed through the following conditional weighted sum performed over the couples of the red black tree that are easy to retrieve in the red black tree

  $$\sum_{(value,count),\ value>c} count \cdot (value - c)$$

  more generally, the red black tree will allow to easily compute sums of the following form

  $$\sum_{p=p1}^{p2} \max\{0, w(p) - k\} \text{ where } k \text{ is a (pre-computed) constant}$$
- the tree attached to each node can easily be derived by cloning the tree of the previous node in the vehicle route and increasing the capacity reached at the current node
- the tree representation is $O(\text{n})$ in space

The coding of the red-black tree for each node of the previous is represented in Table 4. The table shows the count for each node (columns) and possible value (lines), so the set of couples of the red-black tree at position m is formed by assembling columns m and *content*.

Table 4: Matrix representation of the Red-Black trees for each node

node content	0	1	2	3	4	5	6	7
−1	0	0	0	0	1	1	1	1
0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0
2	0	0	0	1	1	1	1	1
3	1	1	1	1	1	1	1	1
4	0	1	1	1	1	1	1	1
5	0	0	0	0	0	1	1	2
6	0	0	1	1	1	1	2	2

For example, to compute the total violation of a route, we consider the last node of the route and content strictly above 4, this result in $2.(6-4)+2.(5-4) = 6$ as already computed before.

An additional benefit is that the contribution of a segment s_1 s_2 can also be computed by combining information of the couples $s_1 - 1$ and s_2. For example, for the segment $5 \rightarrow 6 \rightarrow 7$, overhead surface between 5 and 7 is $(2-1).(6-4)+(2-0).(5-4) = 4$. Note that the count information is directly factorised in this computation.

Considering the different neighbourhood operations, efficient violation update properties can be defined for inserting, removing, moving, flipping a segment. Those are given in annex. All those formula are composed of terms that can efficiently be evaluated using the pre-computations described above and based on the red-black tree structure, i.e. all summation can be reduced to an evaluation on the end node and the node preceding the start of the segment. However those computations cannot be efficiently maintained over complex explorations without our proposed bijection management. In the next section we will show that the above formulation can fit our template, then we will compare the resulting implementation with the above ad-hoc implementation.

4.3 Template Instantiation

We use here the conventions defined in Section 2.3. Pre-computations associate to a node are:

- $cont[n]$ content of vehicle m at node n
- $rb[n]$ the associated red-black tree (as defined in Section 4.2)

Given that, it would be very expensive to really fully compute the + and − on red-black tree. In order to be efficient, different specialised types based on the same abstract type are defined and used in specific contexts. Each specialised type contains different information and the + and − operators work with subclasses as needed, considering how they are used as summation of differences (so minus operators are connected with the global summation). Table 5 shows our template instantiated for the capacity global invariant. Note that for the integrals $+\infty$ et $-\infty$ respectively represents the biggest and smallest vehicle content.

Table 5: Template instantiation for the capacity global invariant

Template	**Definition**
T	Actually an abstract type, concrete types are: T_1, T_2, T_3 Those are used in specific context for optimizing performance
T_1	$(node : Int, prevNode : Int, content : Int, rb : RedBlackTreeMap[Int])$
T_2	$(fromRB : RedBlackTreeMap[Int], toRb : RedBlackTreeMap[Int],$ $deltaContent : Int, contFrom : Int)$
T_3	$(viol : Int, contEnd : Int)$
$- : T_1 \times T_1 \times flip$ $\rightarrow T_2$	$(n_2, n_2 - 1, cont[n_2], rb[n_2]) - (n_1, n_1 - 1, cont[n_1], rb[n_1])$ $\rightarrow \begin{cases} (rb[n_1], rb[n_2], cont[n_2] - cont[n_1], cont[n_1]) & \text{if}\neg\text{flip} \\ (rb[n_2 - 1], rb[n_1 - 1], cont[n_2] - cont[n_1], cont[n_2]) & \text{if flip} \end{cases}$
$+ : T_3 \times T_2 \times flip$ $\rightarrow T_3$	$(viol, contAtEnd) + (fromRB, toRB, deltaCont, contFrom)$ $\rightarrow \begin{cases} (viol + \int_{c-(contEnd-contFrom)}^{+\infty} toRB - fromRB, contEnd + deltaCont) & \text{if}\neg!\text{flip} \\ (viol + \int_{-\infty}^{contEnd+contFrom-c} toRB - fromRB, contEnd + deltaCont) & \text{if flip} \end{cases}$
$value : node \rightarrow T_3$	$(\max\{(\sum_{i=0}^{n} w(i)) - c, 0\}, \sum_{i=0}^{n} w(i))$

In short, for a segment moved, the − receives the content and red-black trees associated with the ends. It sends to the + operator both red-black and the content difference at the end taking into account whether the segment was flipped or not.

In addition to the − output, + also receives the vehicle violation and its content for the position just before the new position of the moved segment. For this, the integral is computed using the two red-blacks (always taking into account whether the segment is flipped or not). We only do this at that time because we need to know the contents of the vehicle before the new position of the segment. Knowing the result of the integral, the vehicle violation can be computed and the vehicle content can also be updated.

4.4 Benchmarking

The goal of our benchmarking is to assess the algorithmic efficiency of our framework on this specific case. We can actually rely on three implementations:

- a naive non-incremental implementation
- an ad-hoc implementation using pre-computation but without reuse capability (no bijection)
- a fully fledge implementation based on the instantiated template presented in Section 4.3

Two kind of neighbourhood were used: simple ones and cross products. Considering first simple neighbourhoods, Table 6 shows that all operations are performing better with a gain between 2.3 (for switch between route with requires double update) and 15 (for simple insert).

However considering cross-products neighbourhoods, this gain is totally lost in the ad-hoc implementation, resulting in similar times as the naive implementation (e.g. about 5s for a routing problem of 30 nodes).

When using updates through the bijective function, the efficiency is restored at roughly the same level of simple neighbourhoods provided lazy updates are used to update the bijection (i.e. not necessary at the leaf node shown in Figure 3). Although we have not yet fully benchmarked the behaviour over a large set of problems, with expect our solution to keep close to linear as a function of the number of routed node and neighbourhood involved.

Table 6: Comparison of main operations at route start (route with 2000 nodes)

Operation	Naive	Pre-computations	Gain
Insert	1.5s	0.1s	15
Remove	1.0s	0.1s	10
Flip	1.0s	0.2s	5
Move	0.9s	0.4s	2.3

5 Conclusion and Perspectives

In this paper, we extended the basic mechanisms provided by our sequence variable to manage global constraints by defining an abstract invariant template that makes their implementation much easier. We also contributed efficient mechanics to manage the execution of the pre-computation and the queries to the pre-computation. Pre-computation being an expensive operation, we designed our framework so that it can exploit pre-computation even if the sequence was modified since it was performed through a specific bijective function. We illustrated the use of our template on a complex capacity constraint.

As future work, we plan to capture a wider variety of constraints to further assess and refine our template as required. We will also carry out a wider scale benchmarking campaign based on a larger set of examples and also compare our work with global constraints implemented in other engines not only from the performance point of view but also for the ease of development and evolution.

Acknowledgement

This work is partly funded by the ePick Innovation Partnership Project from the Walloon Region (grant nr. 7570).

References

1. Glover, F.W., Kochenberger, G.A.: Handbook of Metaheuristics. International Series in Operations Research & Management Science. Springer US (2003)
2. Mladenović, N., Urošević, D., Hanafi, S.: Variable neighborhood search for the travelling deliveryman problem. 4OR **11** (2013) 57–73
3. Savelsbergh, M.W.P.: The vehicle routing problem with time windows: Minimizing route duration. ORSA journal on computing **4** (1992) 146–154
4. OscaR Team: OscaR: Operational research in Scala (2012) Available under the LGPL licence from `https://bitbucket.org/oscarlib/oscar`.
5. De Landtsheer, R., Ponsard, C.: Oscar.cbls : an open source framework for constraint-based local search. In: Proceedings of ORBEL'27. (2013)
6. De Landtsheer, R., Guyot, Y., Ospina, G., Germeau, F., Ponsard, C.: Reasoning on sequences in constraint-based local search frameworks. In: Proc. of the 15th Int. Conf. on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research, CPAIOR (accepted). (2018)
7. De Landtsheer, R., Guyot, Y., Ospina, G., Ponsard, C. In: Combining Neighborhoods into Local Search Strategies. Springer International Publishing, Cham (2018) 43–57
8. De Landtsheer, R., Germeau, F., Guyot, Y., Ospina, G., Ponsard, C.: Easily building complex neighbourhoods with the cross-product combinator. In: Proceedings of the 32th ORBEL Annual Meeting. (2018) 127 – 128
9. De Landtsheer, R., Guyot, Y., Ponsard, C.: A high-level, modular and declarative modeling framework for routing problems. In: Proceedings of ORBEL'28. (2014)
10. Ponsard, C., De Landtsheer, R., Guyot, Y., Roucoux, Françoisand Lambeau, B.: Decision making support in the scheduling of chemotherapy coping with quality of care, resources and ethical constraints. In: Proc. of the 19th Int Conf. on Enterprise Information Systems (ICEIS), Porto, Portugal, April 26-29. (2017)
11. Meurisse, Q.: Development of a Capacity Global Constraint for Vehicle Routing. Internship report (in French), University of Mons (2017)

Annex - Update of violation after neighbourhood operators

The following table gives efficient computation for updating vehicle violation after specific neighbourhood operators. The full proofs are not reproduced here for space reason but they are available in [11].

Table 7: Incremental update of violation after specific neighbourhood operators

Move type	Violation delta
Inserting a node of weight w_{news} after node p^*	$+\sum_{p=p^*+1}^{w^*} \max\left\{0, content(p) - (c - w_{new})\right\}$ $-\sum_{p=p^*+1}^{w^*} \max\left\{0, content(p) - c\right\}$ $+\max\{0, content(p^*) + w_{new} - c\}$
Removing node p^* of weight w_{news}	$+\sum_{p=p^*+1}^{w^*} \max\left\{0, content(p) - (c + w_{news})\right\}$ $-\sum_{p=p^*}^{w^*} \max\left\{0, content(p) - c\right\}$
Moving a segment $p_1 \to \cdots \to p_m$ after $p^* > p_m$	$+\sum_{p=p_m+1}^{p^*} \max\left\{0, content(p) - (c + \sum_{k=p_1}^{p_m} w(k))\right\}$ $-\sum_{p=p_1}^{p^*} \max\left\{0, content(p) - c\right\}$ $+\sum_{p=p_1}^{p_m} \max\left\{0, content(p) - (c - \sum_{k=p_m+1}^{p^*} w(k))\right\}$
Moving a segment $p_1 \to \cdots \to p_m$ after $p^* < p_1 - 1$	$-\sum_{p=p^*+1}^{p_m} \max\left\{0, content(p) - c\right\}$ $+\sum_{p=p_1}^{p_m} \max\left\{0, content(p) - (c + \sum_{k=p^*+1}^{p_1-1} w(k))\right\}$ $+\sum_{p=p^*+1}^{p_1-1} \max\left\{0, content(p) - (c - \sum_{k=p_1}^{p_m} w(k))\right\}$
Moving a segment from route v_1 after node p^* of route v_2 (update for v_2)	$+\sum_{p=p^*+1}^{v_2^*} \max\left\{0, content_{v_2}(p) - (c - \sum_{k=p_1}^{p_m} w(k))\right\}$ $-\sum_{p=p^*+1}^{v_2^*} \max\left\{0, content_{v_2}(p) - c\right\}$ $+\sum_{p=p_1}^{p_m} \max\left\{0, content_{v1}(p) - (c - content_{v_2}(p^*) + content_{v_1}(p_1 - 1))\right\}$
(update for v_1)	$-\sum_{p=p_1}^{v_1^*} \max\left\{0, content(p) - c\right\}$ $+\sum_{p=p_m+1}^{v_1^*} \max\left\{0, content(p) - (c + \sum_{k=p_1}^{p_m} w(k))\right\}$
Flipping segment $p_1 \to \cdots \to p_m$	$-\sum_{p=p_1}^{p_m} \max\left\{0, content(p) - c\right\}$ $+\sum_{p=p_1-1}^{p_m-1} \max\{0, h - content(p)\}$ where $h := content(p_1 - 1) + content(p_m) - c$

A New Hidden Markov Model Approach for Pheromone Level Exponent Adaptation in Ant Colony System

S. Bouzbita[1] A. EL Afia[2] and R. Faizi[3]

ENSIAS - Mohammed V University, Rabat, Morocco
safae.bouzbita@gmail.com[1]
a.elafia@um5s.net.ma[2]
rdfaizi@gmail.com[3]

Abstract. We propose in this paper a Hidden Markov Model (HMM) approach to avoid premature convergence of ants in the Ant Colony System (ACS) algorithm. Indeed, the proposed approach was modelled as a classifier method to control the convergence through the dynamic adaptation of the α parameter that weighs the relative influence of the pheromone. The implementation was tested on several Travelling Salesman Problem (TSP) instances with different number of cities. The proposed approach was compared with the standard ACS and the existing fuzzy logic in the literature. The experimental results illustrate that the proposed method shows better performance.

1 Introduction

The TSP is one of the most complex combinatorial optimization problems studied in computer science, logistics, and transportation industries [1–3]. Where, the task is finding the shortest tour that visits all the cities in a given list once and only once starting from one city and returning to the same city [4].

The TSP is then the optimization problem to find a Hamiltonian cycle that minimizes the length of the tour.

For this minimization mission, $(n-1)!$ possibilities of solutions have to be compared, which make it very hard to solve and then belongs to the NP-hard problem that cannot be optimally in a polynomial time. Many heuristics and meta-heuristics have been proposed to find near optimal solution to it. The Ant Colony Optimization (ACO) meta-heuristic is one of the most powerful algorithms for solving the TSP [5]. Since the development of the first (ACO) algorithm by Dorigo 1991 [6], many other variants have been proposed, which differ one to the other in the Update Pheromone procedure and some characteristics in the Construction Solution. One of the most interesting variants is the Ant Colony System (ACS). The (ACS) algorithm differs in several ways from the other (ACO) techniques. The main steps of ACS algorithm are:

Algorithm 1: Standard ACS

```
1 Initialization phase: Initialize pheromone trails with $\tau_0 = 1/L_{NN}$
2 Construction phase:
  repeat
      foreach ant do
          build a feasible solution according to (1) and (2)
          Local update according to (3)
      end
3     Global update according to (4)
  until stop criteria is reached;
4 Return the best found tour $L_{best}$
```

- **Initialization:** In this step, the parameters are set and pheromone matrix is initialized with a small values.

- **Construction solution:** ants uses the so called pseudo-random proportional rule to construct a feasible solution in the case of TSP a complete tour: with a probability q_0 the next city is chosen as

$$argmax_{u \in J_k(r)}[\tau(r,u)]^{\alpha}[\eta(r,u)]^{\beta} \quad ifq \leq q_0 \tag{1}$$

and with probability (1-q_0) the random proportional rule is used as

$$p^k_{rs} = \begin{cases} \dfrac{[\tau(r,s)]^{\alpha}.[\eta(r,s)]^{\beta}}{\sum_{u \in J_k(r)}[\tau(r,u)]^{\alpha}[\eta(r,u)]^{\beta}} & if \quad s \in J_k(r) \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where $\tau(r,s)$ and $\eta(r,s)$ are the pheromone level and heuristic value between i and u, and $J_k(r)$ is the set of component solutions yet to be chosen by ant k positioned on r. The state transition rule resulting from (1) and(2) is called pseudo-random-proportional rule.
A local pheromone update rule is applied during the solution construction of the ants. Each time an ant moves to the next city j the amount of pheromone between (r,s) is modified according to:

$$\tau(r_k, s_k) := (1-\xi)\tau(r_k, s_k) + \xi\tau_0 \tag{3}$$

where $\xi \in (0,1)$ is a parameter called local pheromone parameter, and τ_0 is the value that initializes the trails of pheromone, τ_0 is a very small constant with value $\frac{1}{n.L_{nn}}$, where L_{nn} is the length of a nearest neighbour tour and n is the number of instances in the problem. The aim of the local update pheromone is avoiding the stagnation by decreasing the pheromone value on the used edges and make them less attractive.

- **Global pheromone update:** At the end of each iteration the amount of pheromone is updated again according to either the iteration-best or the global so far solution as

$$\tau(r_k, s_k) := (1-\rho)\tau(r_k, s_k) + \frac{\rho}{(L_{best})} \tag{4}$$

This rule applied after all ants have constructed their solutions.

- **Stop criterion:** The algorithm can be stopped after it reached the stop criterion which is a maximum number of iterations without improvement and returns L_best.

The performance of ACO algorithms depends strongly on the given values to parameters. In the earliest ACO applications, parameters values are set constant during the running of the algorithm. However, modifying the values of parameters throughout the run of the algorithm can evolve the performance of algorithm. Parameter adaptation is becoming a considerable task in the field of evolutionary algorithms (EAs). The adaptation of parameters while running the algorithm using machine learning techniques is also the influential theme in the research area.
Several strategies have been proposed in the literature for adapting parameters while solving a problem. For example, in [7–12] authors have chosen the HMM machine learning to adapt parameters of other meta-heuristics at runtime.
In this paper, we propose the machine learning technique Hidden Markov Model (HMM) to adapt the exponent of pheromone level α parameter dynamically according to some performance measures while solving some TSP problems.
The rest of the paper is organized as follows. Section 2 presents the relate work. Section 3 describes a new method of parameter adaptation through Hidden Markov Model. Section 4 outlines the experimental results and comparison. Finally, Section 5 gives the conclusion.

2 Related Work

The exponent on the pheromone level α of ACO algorithm does not received enough attention in the literature, the reason why Meyer [13] decided to study the influence of this parameter on the ACO performance by proposing an algorithm called critical cycle Ant System (ccAS) in which the subsequent runs start from an already developed pheromone matrix, and he proved that the parameter α determine the quality of the convergence of ACO also it rends the ant algorithms more efficient.

In the same context, Neyoy et al [14] proposed a new approach to control the diversity in ACO by the dynamic variation of α parameter using a convergence fuzzy logic controller that aims to avoid or slow down full convergence.

In [15] Chusanapiputt et al proposed an Selective Self-Adaptive approach Ant System (SSAS) to solve the constrained unit commitment problem by adapting transition probability parameters and population size through increasing and decreasing the value of $\alpha and \beta$ and cooperation with Effective Repairing Heuristic Module (ERHM) and Candidate Path Management Module (CPMM).

Li et al [16] proposed an information entropy based approach to solve the premature convergence problem of the ACO algorithm, which is applied to tuning α and β parameters on Traveling Salesman Problem.

Martens et al [17] suggested a a self-adaptive implementation of Max-Min Ant System (MMAS) applied to the data mining field for extracting rule-based classifiers. The aim of their called AntMiner+ classification technique, is to find suitable values for α and β parameters by creating a new vertex group in the construction graph for each parameter. The values of α and β are limited to integers 1,2 and 3.

Khichane et al. [18] proposed a self-adaptive approach to modify the parameters α and β of MMAS by introducing two reactive frameworks that differ in he granularity of parameter learning and applying it to constraint satisfaction problems. The updated parameters are considered independent of each other. For this adaptation authors developed two mechanism. The first one, called global parameter learning ant-solver (GPL-Ant-solver) define one common parameter for all the colony during the construction solution by ants. In the second mechanism, called distributed parameter learning ant-solver (DPL-Ant-solver), the values of α and β are updated at each step of the construction solution.

Ling et al [19] introduced the Artificial Fish Swarm Algorithm (AFSA) to solve the parameter modification problem of ACO algorithm when applying it to the TSP. Three parameters were updated $\alpha, \rho and Q$. In their work, authors defined the same parameter setting for all ants.

Melo et al [20] proposed a multi-colony ACS algorithm, where various ant colonies try to solve the same problem at the same time. Each colony has its proper values of parameters $\alpha, \beta, \rho and q_0$. The main idea of their work is to replace the parameters of the worst colony by the values of the same parameters of the best colony.

For their part, Olivas et al [21] developed a dynamic parameter adaptation approach for Ant Colony Optimization (ACO) based on interval type-2 fuzzy systems, for controlling the ability of exploration and exploitation. The objective of their method is to be able to apply it to a wide range of problems without the need of finding the optimal values of parameters for each problem. The exponent on the pheromone level α and the rate of evaporation of the pheromone trail ρ are the chosen parameters to be change.

3 Proposed method

In this section a new improved ACS algorithm based on HMM method is proposed to adapt the exponent on the pheromone level α according to some performance measures. The reason why we have chosen to tune this parameter, is its strong influence on the quality of convergence and performance of the ACS algorithm.

The main idea of this work is that the value of α should be modified according to the diversity throughout the population and the closeness to the best known solution which are considered as the performance measures. For example, if the diversity throughout the population is high so there are several different paths explored by the ants, this means we don't require any more exploration and we need to exploit the accumulated information by setting the α parameter to a high value. Considering the error from getting the best solution, if the error is high this means we are far from the best solution so we need this time to explore more solutions by setting α to a low value.

To describe the diversity throughout the population we have chosen the Variance measure which has the power to indicates how far ants are spread out.

$$Variance = \frac{\sum_i^m (L_i - \mu)^2}{m} \tag{5}$$

Where, L_i is the length of the found solution by ant i, μ is the mean of all solutions found by the population, and m is the number of ants. For the closeness to the best known solution the error variable was chosen as a measure performance.

$$Error = L_{best} - L_{best-known} \tag{6}$$

Where, L_{best} is the best found solution by the population, and $L_{best_{k}nown}$ is the best known solution for [TSPLIB].
These two measures are considered as observations for the HMM, while the values of α are considered as states.

3.1 Hidden Markov Model

Hidden Markov Model is a learnable stochastic automate which consists of two stochastic processes. The first process is an underlying unobservable Markov Chain that externally cannot be visible (hidden) and characterized by states and transition probabilities. The second one is an observable process producing observation symbols depending on probability distribution of the first process. In practice, A HMM noticed by the symbol λ.
We now define the five following elements for the proposed model:

- $S = \{L, ML, M, MH, H\}$ is the hidden states set
- $V = \{LL, LM, LH, ML, MM, MH, HL, HM, HH\}$ is the observation symbols per state set
- $\Pi = [\pi_1, \pi_2, \pi_3, \pi_4, \pi_5] = [1, 0, 0, 0, 0]$ is the initial probability where π_i is the probability of being in the state S_i. Where, at time t=0 the model is in state "Low" for α parameter. As it mentioned before, at the beginning of the algorithm we need to explore maximum area that's why we have to set the α parameter at a low value.
- $A = [a_{ij}]$ is a matrix of transition probabilities from state S_i at time t to state S_j at time t+1. In our method, we considered a five state left-to-right HMM to describe the ants states. In the case of left-to-right HMM, the initial state probability is equal to one. Thus, all the ants start with the same state.

- $B = [b_{jk}]$ is the emission matrix of observing a symbol V_k from a state S_i. The emission matrices were defined according to some knowledge that a low value of variance means that the ants may be stuck in an optimal solution so we need to explore more solutions by decreasing the value of α. And a low value of error means that the ants are close to the best known solution so we have to exploit the accumulated information by increasing the value of α.

$$A = (a_{ij}) = \begin{bmatrix} 0.5 & 0.5 & 0 & 0 & 0 \\ 0 & 0.5 & 0.5 & 0 & 0 \\ 0 & 0 & 0.5 & 0.5 & 0 \\ 0 & 0 & 0 & 0.5 & 0.5 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad B = (b_{jk}) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0.5 & 0 & 0 & 0 & 0.5 & 0 \\ \frac{1}{3} & 0 & 0 & 0 & \frac{1}{3} & 0 & 0 & 0 & \frac{1}{3} \\ 0 & 0.5 & 0 & 0 & 0 & 0.5 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

We have defined five states corresponding to the values of α such as: Low (L), Medium Low (ML), Medium (M), Medium High (MH), High (H). For the observation symbols we have concatenated the performance measures Variance and Error such as each measure represented by three symbols L, M, H. Where, L means Low, M means Medium, and H means High. So we obtain nine possible concatenation.
In our method, the Viterbi algorithm was adopted so as we tried to compute the most likely sequence of states that produced the current observation sequence. The dynamic adaptation of the parameter q_0 is done according to the last state in the found sequence of states. For example, if the most likely state sequence for α parameter is H, L, M, MH, L, ML, M, then its value is updated according to the last state which is "Medium". In addition to the Viterbi, we have used the well known Baum-Welch training method to adjust the HMM parameters $\lambda = (A, B, \pi)$ during the run time.

Adjusting HMM parameters and State determination At the end of each iteration we perform an Online Learning for HMM parameters by the Baum-Welch algorithm. The Baum-Welch algorithm uses the expectation maximization algorithm to find the maximum likelihood estimate of the HMM parameters given a sequence of observed data. After that, to find the most likely explanation or the most likely state that generated a particular sequence of observations the Viterbi algorithm is used to find a maximum over all possible state sequences. This algorithm has the ability to deal with the entire sequences of hidden states from the beginning of the algorithm till the current iteration,and then make a decision base on the whole history, which makes it advantageous compared to other algorithms that depend only on the information of the current iteration.

Algorithm 2: Parameters estimation and state determination

```
Input: O=$o_1$,$o_2$,...,$o_T$, S, $\lambda = (A, B, \pi)$
repeat
    Re-estimate $\lambda$ using Baum-Welch
    Find the most likely sequence of states $S_T$ using Viterbi
until no increase of $P(O/\lambda)$ or no more iterations are possible to do;
```

Proposed algorithm In the proposed algorithm, each ant builds a tour by choosing the next node using the pseudo random-proportional action choice rule.

Algorithm 3: The pseudocode of our proposed algorithm

```
Initialization:
Construction Solution:
while termination condition not met do
    foreach ant k in the population do
        chose the next according to (1) and (2)
        local pheromone update according to (3)
    end
    Update the global best solution;
    Global pheromone update according to (8)
    Compute Variance and Error according to (5) and (6)
    if $0 < Variance \leq \frac{MaxVariance}{3}$ then Variance = L;
    else if $\frac{MaxVariance}{3} < Variance \leq \frac{2 * MaxVaraince}{3}$ then Variance = M;
    else Variance= H;
    if $0 < Error \leq \frac{MaxError}{3}$ then Error= L;
    else if $\frac{MaxError}{3} < Error \leq \frac{2 * MaxError}{3}$ then Error= M;
    else Error= H;
    Return O=Variance Error; Apply algorithm 2 to find the suitable state
    update $\alpha$ according to the found state,
    if state= L then $\alpha = 0.6$;
    if state= ML then $\alpha = 0.7$;
    if state= M then $\alpha = 0.8$;
    if state= MH then $\alpha = 0.9$;
    else
        $\alpha = 1$
    end
end
Return $L_{best}$
```

After building a tour the variance and the error calculated and converted into symbols according to the determined intervals. The symbols then combined to build an observation, then the observation sent to the Viterbi algorithm to determine which state is the most likely responsible for producing this observation. The sequence of observation is incremented after each iteration, so the number of elements of a sequence equal to the number of iterations.

4 Experimental results and comparison

To test the efficiency of the proposed algorithm, we compared it with the standard ACS algorithm also with Fuzzy Logic results.
The algorithm was run according to the best known values of ACS algorithm parameters [22] which are $\beta = 2, \rho = 0.1$, $q_0 = 0.9$, and m=10. The initial position of ants is set randomly on all experiments.
The TSP benchmark instances used in this study were chosen from the TSPLIB [23] according to the most common used instances in the literature. The algorithm is developed on MATLAB. Each instances was ran for 1000 iterations. The stop condition is: 200 iteration as the maximum number without improving in results.

4.1 Comparison on the solution accuracy

Table 1 shows the results of running the proposed and the standard ACS algorithms on the chosen TSP instances.
The meaning of each column of the table is as follows.

- The first column represents the names of the chosen TSP instances.
- The second column represents the best found solution by the standard ACS algorithm.
- The third column represents the amount of time used for the standard ACS processing.
- The fourth column represents the best found solution by the proposed method ACSHMM.
- The fifth column represents the amount of time used for the proposed method.
- The sixth column represents the best found solution by Fuzzy Logic [25].

Table 1. Summary of results of the standard ACS and ACSHMM algorithms on some TSP instances

Problem	standard ACS		ACSHMM		FuzzyLogic	best known solution
	Solution	CPUtime	Solution	CPUtime	Solution	
Eil51	435.4 [104]	143	428.8 [325]	201	431.9 [199]	426
St70	690 [224]	252	677 [246]	292.6	691 [66]	675
Eil76	566 [12]	167.7	555.79 [286]	275.79	556 [479]	538
Rat99	1228.5 [614]	1158.4	1221 [403]	711.8	1234.2 [315]	1211
Eil101	675.7 [7]	316.78	656.4 [75]	363.3	655.6 [218]	629
Lin105	14551 [165]	698.8	14474 [148]	436.79	14543.1 [27]	14379
Pr107	44620 [82]	403.6	44481 [127]	510.5	45757.9 [329]	44303
Pr124	59632.9 [25]	472.2	59159 [309]	1101.5	60203.3 [322]	59030
Rat195	2458.8 [114]	1833.3	2434 [28]	1400	2401.9 [65]	2323
pr264	53118.4 [101]	5590	52514.5 [31]	2790	53429.6 [78]	49135

The numbers between brackets are the numbers where the algorithms found the best solution.
From the table 1 we can observe that the proposed algorithm ACSHMM gives better results in the convergence speed and the solution accuracy compared with both the standard ACS and the proposed one in the literature by fuzzy logic. Thus in The most instances our proposed algorithm outperforms the others except in one instance which is considered as a small problem. We can see that the found solutions by the proposed ACSHMM method are very close to the best known

solutions.
Also, the important aspect about our proposed algorithm is its behaviour under bad parameter setting. Thus, the optimal known value for the number of ants equal to the size of the TSP instance, but when we set the number of ants to 10, our proposed algorithm outperforms the standard and the Fuzzy Logic algorithms.

4.2 Comparison on the convergence speed

In order to show the convergence speed of the proposed method we calculated its CPUtime (Table 1), also the charts that show the progress of the tour length with the iteration number have been drawn.
The first chart correspond to St70.tsp instance, shows that the proposed method converge to a better solution, as we can observe from the table 1, even if the standard ACS found its best solution in the 82^{th} iteration but the proposed method found better one.

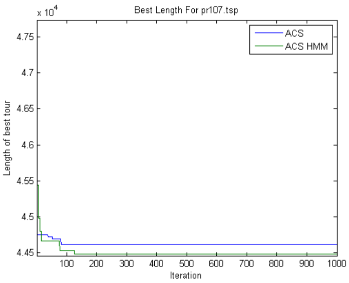


Fig. 1. Sample run on pr107.tsp

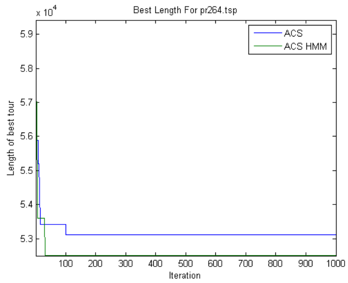


Fig. 2. Sample run on pr264.tsp

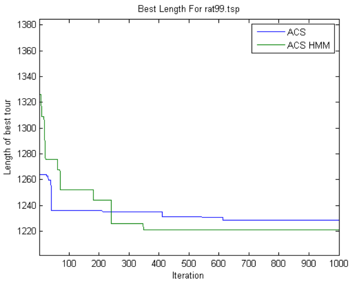


Fig. 3. Sample run on rat99.tsp

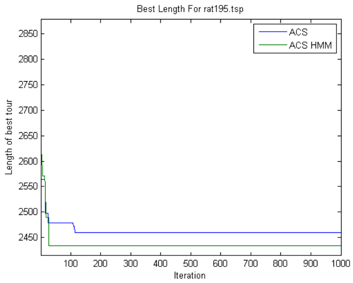


Fig. 4. Sample run on rat195.tsp

In the second, the third and fourth charts, the proposed method can achieved better solutions and can made faster convergence.
From those charts and table we can assume that the results of our proposed method are very encouraging.

4.3 Statistical test

We used the Wilcoxon Rank Test in a pair-wise comparison procedure under significance level $\alpha = 0.05$ as a statistical test to compare the methods. The reason why we have used the Wilcoxon Rank Test is its consideration to the quantitative differences in the algorithms performance. Also, it is the recommended statistical test method used in many other researches [26, 27]
The null hypothesis says that the found solutions of the ACSHMM method are worse than the ACS method, while the alternative hypothesis says that the solutions of the ACSHMM method

Table 2. Statistical validation for the TSP benchmark instances with ACSHMM as control algorithm

TSP	Eil51	St70	Eil76	Rat99	Eil101
Standard ACS	2.57E-02	3.445E-01	1.04E-02	6.5E-02	9.09E-01
TSP	Lin105	Pr107	Pr124	Rat195	pr264
Standard ACS	2E-02	2.39E-02	4.69E-02	4E-02	5.8E-03

are better when compared with the standard ACS.
From the Table 2 that represents the p-value for the test, we can observe that our proposed algorithm outperforms the original ACS with level of significance of 5%. As we can notice, the calculated p-value is below the significance level in most benchmark instances. Also, we can see that the proposed method only in one instance fails to reject the null hypothesis, however in all other results our proposed method converge to better solutions when compared to other methods.

5 conclusion

We can conclude from proposing the Hidden Markov Model (HMM) controller to the Ant Colony System (ACS) algorithm for the sake of dynamic adaptation to its exponent of pheromone level α parameter when applied to some Travelling Salesman Problems (TSP) that this proposed algorithm could improve the quality of solutions compared with the standard algorithm and the proposed one by the Fuzzy Logic on one hand. On the other hand, we can see that the proposed method converge faster to better solutions in all most instances. Also, from the results of the statistical tests we can observe that only in one instance the test fails to reject the null hypothesis, however in all other results the proposed approach found enough evidence to reject the null hypothesis with a level of significance of 5%. Thus, the ACSHMM outperforms the other methods in both solution accuracy and speed convergence.

References

1. Erol, Ayşe Hande, Merve Er, and Serol Bulkan. "Optimizing the ant colony optimization algorithm using neural network for the travelling salesman problem." Actas de la Conferencia Internacional de. 2012.
2. Helsgaun, K., and Ngassa, J. L. ACO and TSP.(2007).
3. Matai, R., Singh, S., and Mittal, M. L. Traveling salesman problem: an overview of applications, formulations, and solution approaches. In Traveling salesman problem, theory and applications. InTech.(2010).
4. Diaby, M. The traveling salesman problem: a linear programming formulation. arXiv preprint cs/0609005.(2006).
5. Filip, Exnar, and Machac Otakar. "The travelling salesman problem and its application in logistic practice." WSEAS Transactions on Business and Economics 8.4 (2011): 163-173.
6. Dorigo, M., Maniezzo, V., and Colorni, A. The ant system: An autocatalytic optimizing process.(1991).
7. Bouzbita, S., El Afia, A., Faizi, R., and Zbakh, M. (2016, May). Dynamic adaptation of the ACS-TSP local pheromone decay parameter based on the Hidden Markov Model. In Cloud Computing Technologies and Applications (CloudTech), 2016 2nd International Conference on (pp. 344-349). IEEE.El
8. Bouzbita, S., El Afia, A., and Faizi, R. (2016, September). A novel based Hidden Markov Model approach for controlling the ACS-TSP evaporation parameter. In Multimedia Computing and Systems (ICMCS), 2016 5th International Conference on (pp. 633-638). IEEE.
9. Aoun, O., Sarhani, M., and El Afia, A. (2016). Investigation of hidden markov model for the tuning of metaheuristics in airline scheduling problems. IFAC-PapersOnLine, 49(3), 347-352.
10. Aoun, O., Sarhani, M., and El Afia, A. (2018). Hidden markov model classifier for the adaptive particle swarm optimization. In Recent Developments in Metaheuristics (pp. 1-15). Springer, Cham..
11. El Afia, A., Sarhani, M., and Aoun, O. (2017). Hidden markov model control of inertia weight adaptation for Particle swarm optimization. IFAC-PapersOnLine, 50(1), 9997-10002.
12. Lalaoui, M., El Afia, A., and Chiheb, R. (2016, September). Hidden Markov Model for a self-learning of Simulated Annealing cooling law. In Multimedia Computing and Systems (ICMCS), 2016 5th International Conference on (pp. 558-563). IEEE.
13. Meyer, B. Convergence control in ACO. In Genetic and Evolutionary Computation Conference (GECCO), Seattle, WA, late-breaking paper available on CD.(2004, June)

14. Neyoy, H., Castillo, O., and Soria, J. Fuzzy logic for dynamic parameter tuning in ACO and its application in optimal fuzzy logic controller design. In Fuzzy Logic Augmentation of Nature-Inspired Optimization Metaheuristics (pp. 3-28). Springer International Publishing. (2015)
15. Chusanapiputt, S., Nualhong, D., Jantarang, S., and Phoomvuthisarn, S. Selective self-adaptive approach to ant system for solving unit commitment problem. In Proceedings of the 8th annual conference on Genetic and evolutionary computation (pp. 1729-1736). ACM.(2006, July).
16. Li Y., LiW.: Adaptive ant colony optimization algorithm based on information entropy: Foundation and application. Fundamenta Informaticae 77(3):229–242 (2007)
17. Martens, D., De Backer, M., Haesen, R., Vanthienen, J., Snoeck, M., and Baesens, B. Classification with ant colony optimization. IEEE Transactions on Evolutionary Computation, 11(5), 651-665 (2007)
18. Khichane M., Albert P., Solnon C.: An ACO-based reactive framework for ant colony optimization: First experiments on constraint satisfaction problems.In: Stutzle T. (ed.) Learning and Intelligent Optimization, Third International Conference, LION 3, Lecture Notes in Computer Science, vol. 5851, Springer, Heidelberg, Germany, pp. 119–133 (2009)
19. Ling W., Luo H.: An adaptive parameter control strategy for ant colony optimization.In: CIS'07: Proceedings of the 2007 International Conference on Computational Intelligence and Security, IEEE Computer Society, Washington,DC, pp. 142–146 (2007)
20. Melo L., Pereira F., Costa E.: MC-ANT: A multi-colony ant algorithm. In:Artificial Evolution - 9th International Conference, Evolution Artificielle, EA 2009, Lecture Notes in Computer Science, vol. 5975, Springer, Heidelberg, Germany, pp. 25–36 (2009)
21. Olivas, F., Valdez, F., Castillo, O., Gonzalez, C. I., Martinez, G., and Melin, P. (2017). Ant colony optimization with dynamic parameter adaptation based on interval type-2 fuzzy logic systems. Applied Soft Computing, 53, 74-87.
22. Stützle, T.; López-Ibánez, M.; Pellegrini, P.; Maur, M.; de Oca, M. M.; Birattari, M.; Dorigo, M. Parameter adaptation in ant colony optimization. In *Autonomous search.* Springer Berlin Heidelberg.,2012,191-215.
23. Reinelt, G. ”Tsplib discrete and combinatorial optimization.” (1995).
24. Gomez-Cabrero, D.; D. N. Ranasinghe. Fine-tuning the ant colony system algorithm through particle swarm optimization. In: *Proceedings of the International Conference on Information and Automation.*, 2005.
25. Amir, C., Badr, A., and Farag, I. (2007). A fuzzy logic controller for ant algorithms. Computing and Information Systems, 11(2), 26.
26. LaTorre, A., Muelas, S., and Peña, J. M. A comprehensive comparison of large scale global optimizers. Information Sciences, 316, 517-549 (2015).
27. Veček, N., Črepinšek, M., and Mernik, M. On the influence of the number of algorithms, problems, and independent runs in the comparison of evolutionary algorithms. Applied Soft Computing, 54, 23-45 (2017).

An algorithm based on dimensionality reduction through parameterized curves to solve a class of non-convex global optimization

RAHAL Mohamed[a,*], ELLAIA Rachid[b]

[a]*Laboratory of Fundamental And Numerical Mathemativs (LMFN), Department of Mathematics, University of Sétif 1, Sétif, 19000, Algeria*
[b]*Laboratory of Study and Research for Applied Mathematics, LERMA, Mohammed V University of Rabat, Mohammadia School of Engineers, BP. 765, Ibn Sina Avenue, Agdal, Rabat, Morocco*

Abstract

Not so long ago, to develop numerical methods to determine the solution of a non-linear, non-convex and non-differentiable optimization problem that can respond to a set of constraints, also non-linear and non-convex, seemed very difficult. Although the classical theory of optimization can not be applied directly in global optimization problems, traditional tools such as convex analysis, are widely used in the construction of global optimization methods. This approach is an important part of overall deterministic optimization. For example, a remarkable progress has been made in the construction of minimization algorithms of concave functions in convex regions, and also the minimization of the difference functions of two convex functions. The major disadvantage of these algorithms is their inability to answer both questions: see if the solution obtained is global. Global optimization methods were introduced to try to satisfactorily answer the two previous questions. In recent years, much work has been done on the overall optimization of Lipschitz functions of one or more variables. Recently, works on the extension of Piyavskki's method to problems of minimization of less regular functions was presented. It is a question of find-

[*]Corresponding author
Email address: mrahaldz@gmail.com (RAHAL Mohamed)
URL: www.elsevier.com (ELLAIA Rachid)

ing the global minimum of the Hölderian functions of parameters $h > 0$ and $\beta < 1$ defined on a closed and bounded hyper-rectangle Ω. The direct generalisation of this method can be easily extended to the case of several variables, but it is cumbersome and too complicated for practical realisation. For this, we will present a variant of a new deterministic method. This is the Alienor reduction transformation method, developed by Y. Cherruault et al. In this method, we have brought new ideas to apply this approach to problems optimizing Hölderian multivariate functions. The fundamental principle to perform a transformation that reduces the multi-dimensional optimization problem to a one-dimensional problem in order to apply more efficient optimization methods adapted to the case of a single variable. The basis idea is to densify the feasible set by a regular parametric curves ($\alpha - dense$). The Hölder multivariate initial function f of the parameters $h > 0$ and $\beta < 1$ on the hyper-rectangle Ω is transformed into a function of a single variable f^* also Hölderian of parameters $h^* > 0$ and $\beta < 1$ on a closed and bounded real interval. So our problem is reduced to a problem that is easier to solve because there is only one direction to explore. The Alienor method has proven to be very effective by associating with some one-dimensional methods such as covering algorithms. As mentioned in the literature, the standard Piyavskii's method applies to Lipschitz functions $\beta = 1$. The idea consists in constructing an increasing sequence of affine piecewise lower bounding functions, and such that the global minima of the elements of the sequence converges towards the global minimum that we are looking for. This technique can be extended to Hölder functions where the parameters $h > 0$ and $\beta < 1$ are a priori known. Indeed, one builds a sequence of functions having the same properties as the preceding continuation but this time the functions are piecewise parabolic instead of being affine. Our work on this question was based on new results concerning the generation of α-dense curves expressed in a simpler way. The transformation used would consume the global minimizers at least in an approximate way and preserve the properties of the objective function. The algorithm obtained from the coupling of the new variant of Alienor method with the Piyavskii's algorithm is quite simple and relatively effective.

The convergence of the coupled method is proven. We will see that it offers very interesting perspectives to reduce the calculation time. Finally, we will finish our work with a series of numerical applications on test functions given in the literature and having the distinction of having several local minima.

Keywords: Global optimization, Hölder condition, α-dense curves , reducing transformation method, Piyavskii's algorithm.

Metaheuristics for Agent based Intelligent Evacuation System

M. Hajjem[1], H. Bouziri[2], K. Mellouli[3], and E.G. Talbi[4]

[1] LARODEC, ISG, University of Tunis, Tunisia.
hajmanel@yahoo.fr
[2] LARODEC, ESSECT, University of Tunis, Tunisia.
hend.bouziri@gmail.com
[3] LARODEC, IHEC, University of Carthage, Tunisia.
khaled.mellouli@topnet.tn
[4] CRISTAL, Polytech'Lille, Université de Lille1, France.
Talbi@lifl.fr

1 Introduction

Emergency situations are uncertain, and complex for emergency managers. Efficient evacuation plans are the fundamental output of emergency management to reduce damage as much as possible. Traditional evacuation plans include static instructions of guidance based on shortest paths to exits. These plans cannot be adapted to changes in disasters situations. For example, in the fire event, as a fire progresses, some evacuation routes can become blocked by fire or smoke. Smart building technologies can be used to minimize casualties of emergency situations by considering real time evacuation information.

Recently, several intelligent evacuation systems that integrate real time collection and diffusion of buildings' state and dynamic guidance of people. Filippoupolitis and Gelenbe (2009) proposed a distributed system to compute shortest evacuation routes in real-time. The recommended routes are computed by decision nodes and are communicated to the evacuees located in their vicinity. Lujak et al. (2017) considered only the physical distances and the hazard progresses during computing efficient routes. They did not take into consideration crowd congestion on the routes. Lee et al. (2017) proposed a multiple exits evacuation algorithm (MEEA) that guide the occupants in one space to the same exit. The MEEA is based on graph theory and computes a process of exits assignment of the nodes. cite bernardini considered the optimization of dynamic guidance as a bi-level problem. The computation of guidance paths based on the prediction of hazards spread is the upper-level problem. However, the look for evacuation paths based on prediction of congestion is the lower-level problem. A local search procedure is developed to solve the problem.

this paper propose an agent based intelligent evacuation system. This framework defines the interoperability between intelligent management subsystems of smart building. In addition, the agent evacuation guidance optimizer treats the guidance problem as a mult-iobjective problem using metaheuristics.

2 Intelligent Agent Based Evacuation system

We propose an agent based framework for an intelligent evacuation system that combines subsystems of smart buildings in order to provide situation awareness for an evacuation guidance agent that is explained in next section. The framework makes use of three types of agent: Fire control agent, Environment control agent and guidance optimizer agent.

- The first agent is the fire control agent (FCA) wich is related to the different sensors of fire. The FCA integrates an intelligent fire detection algorithm that is able to detect false alarms. Therefore, FCA is apt to obtain more comprehensive and real-time dynamic information about the propagation of the fire.
- The environment control agent (ECA). It consists on a coordinator between agents and building's monitoring subsystems. ECA manage the different monitoring systems such as crowd monitoring system, elevator, doors and windows monitoring system in order to block danger

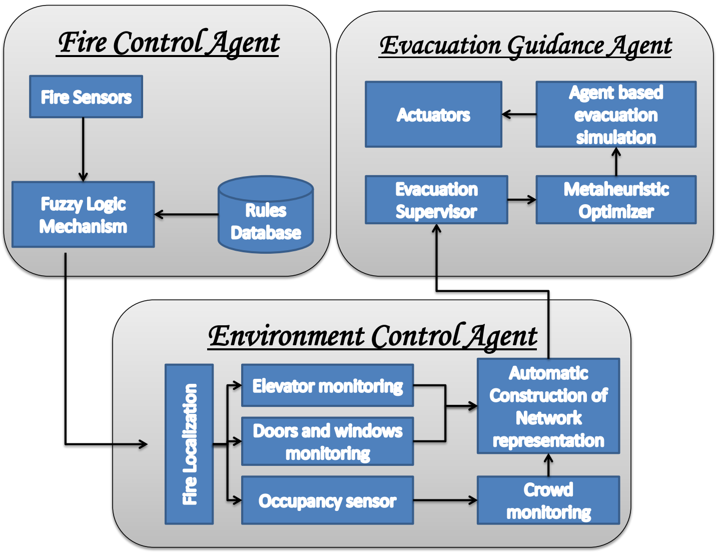


Fig. 1. Proposed Agent based evacuation system framework

paths and analyzes the occupancy distribution. At each unit time this agent, transform the building and collected information into a dynamic navigational map.

- The evacuation guidance agent (EGA) is able to compute dynamic efficient paths for evacuees considering the impact of congestion and fire spread. EGA is a robust and scalable decision-making system that will align the choices of the routes by evacuees with the goal of maximizing the number of saved evacuees as earlier as possible. The main idea of this decision support is the agent based ant colony algorithm.

3 Evacuation Guidance Agent

In our study, we propose an agent based ant colony optimizer to help evacuees to determine evacuate as earlier as possible by optimizing three main objective:

- Maximize safety of paths
- Minimize distance of paths
- Minimize congestion in paths

Considering the assumption that individual follows the group behavior as it was stated by Crano (2000). For example, people prefer to follow others rather than find alternative routes during emergent evacuation if they are not familiar with the building layout. we will treat evacuees departing from the same source as one agent. This agent is characterized by a source node and number of evacuees following this agent. Each agent applies ant colony algorithm in order to look for the best triplet: departure time, nearest exit and efficient path. Therefore, the best solution is sent to the supervisor agent in order to estimate congestion and verify the state of the followed path. Evacuees start to follow selected paths and each unit time they can change their paths depending on the message sent by the supervisor agent.

Bibliography

Ahmed, A. A., Al-Shaboti, M., and Al-Zubairi, A. (2015). An indoor emergency guidance algorithm based on wireless sensor networks. In *Cloud Computing (ICCC), 2015 International Conference on*, pages 1–5. IEEE.

Bernardini, G., Santarelli, S., Quagliarini, E., and D'Orazio, M. (2017). Dynamic guidance tool for a safer earthquake pedestrian evacuation in urban systems. *Computers, Environment and Urban Systems*, 65:150–161.

Crano, W. D. (2000). Milestones in the psychological analysis of social influence. *Group Dynamics: Theory, Research, and Practice*, 4(1):68.

Filippoupolitis, A. and Gelenbe, E. (2009). A distributed decision support system for building evacuation. In *Human System Interactions, 2009. HSI'09. 2nd Conference on*, pages 323–330. Ieee.

Lee, M., Nam, H., and Jun, C. (2017). Multiple exits evacuation algorithm for real-time evacuation guidance. *Spatial Information Research*, 25(2):261–270.

Liu, W. and Qu, C. (2017). Design and optimization of an intelligent evacuation light system. In *Advances in Computer and Computational Sciences*, pages 675–685. Springer.

Lujak, M., Billhardt, H., Dunkel, J., Fernández, A., Hermoso, R., and Ossowski, S. (2017). A distributed architecture for real-time evacuation guidance in large smart buildings. *Comput. Sci. Inf. Syst.*, 14(1):257–282.

Subalakshmi, P., Nandakumar, G., Kumarasan, A., and Vijayakumar, K. (2017). Emergency navigation in rescue with wireless sensor navigation (wsn). *Advances in Natural and Applied Sciences*, 11(6 SI):125–131.

Vector-Quantization Codebook Generation using LBG and Meta-Heuristic Algorithms

Ikram Boubechal[1], Rachid Seghir[1] and Redha Benzid[2]

1. LaSTIC Laboratory, Department of computer Science, University of Batna 2, Algeria
ikram.boubechal@gmail.com, r.seghir@univ-batna2.dz
2. Department of Electronics, University of Batna 2, Algeria
r.benzid@univ-batna2.dz

Keywords: Vector quantization, LBG, meta-heuristics, swarm intelligence, PSO, Firefly, Bat.

1 Introduction

Nowadays, multimedia data applications require an important storage space and large bandwidth. Despite the rapid and continuous progress in storage space, processor speed and numerical communication systems, the need for big space storage and transmission delay time still surpass the existing technologies and become a serious problem. To overcome these weaknesses and reach an optimal exploitation, the compression of multimedia data (image, audio and video) is the most suitable solution. Image compression, based on removing redundant information, reduces the required data bit rate without greatly affecting the image quality [1]. Therefore, several compression methods have been proposed and classified in two categories, lossless and lossy compression. The first one is a non-destructive compression that allows reducing the data size without any information loss. The second category codes the image with quality degradation, since this technique does some loss of data allowing to achieve a higher compression rate than the lossless one. Vector Quantization (VQ) is one of the most successful techniques used in lossy image compression, because of its simple architecture and high compression rate with minimum distortion. VQ allows to generate a limited-length optimal codebook that represents the original image with higher fidelity based on block coding. Furthermore, several algorithms have been proposed in VQ, the well-known method is the Linde Buze Gray (LBG) which is an iterative learning algorithm based on training set [2]. However, the LBG algorithm drawback is the generation of local optimal codebook [3]. In this context, swarm intelligence algorithms are among the meta-heuristic methods that are developed to solve optimization problems by trying to find the most global optimal solution in a short time. In this work, we have used three swarm intelligence algorithms, which are: PSO (Particale Swarm Optimization), Firefly and bat algorithms. The experimental results we have performed demonstrate that the codebook generated by the LBG algorithm is better than the one generated by meta-heuristic methods (PSO, Firefly and Bat) concerning both image quality and generation time.

The rest of the paper is organized as follows: in section 2, an overview of the swarm intelligence algorithms used. The proposed approach is described in section 3. An experimental study is presented in section 4. Finaly section 5 concludes this paper.

2 Swarm intelligence

The main aim of the swarm intelligence is to optimize an objective function, specific to a given problem, in order to find the most optimal solution in a reasonable time. Many algorithms have been proposed in the literature, among them: PSO (Particle Swarm Optimization), Firefly and Bat algorithms.

- The Particle Swarm Optimization (PSO) is a population-based algorithm which uses a swarm of particles; each particle represents a potential solution to the problem of optimization. PSO simulates the feeding behavior of birds in nature, it was initially developed by Kennedy and Eberhart in 1995 [4].

- The Firefly (FF) algorithm was introduced by X.S. Yang in 2008. Its source of inspiration is based on light emission/absorption and mutual attractive behavior between fireflies [5].
- The Bat algorithm (BA) is a recent metaheuristic approach proposed by Yang in 2010 [6]. The idea behind this algorithm is to mimic the echolocation behavior of microbats.

3 Proposed approach

The main purpose of our work is to find the near global codebook for the lossy image problem. Therefore, we have defined the noise peak signal (PSNR) as an objective function. PSNR is a quality measure calculated between the original and compressed images, where a higher value of PSNR indicates a better quality of compression.

The PSNR value is calculated as follows:

$$PSNR = 20\ log_{10}\left(\frac{255}{\sqrt{MSE}}\right) \quad (1)$$

Where MSE (mean square error) is expressed as:

$$MSE = \frac{\sum_{i=0}^{M-1}\sum_{j=0}^{N-1}\left(I(i,j)-\hat{I}(i,j)\right)^2}{MN} \qquad (2)$$

Here I and $\hat{I}$ are the original and compressed images of size M x N, respectively

The objective function is defined as:

$$F^* = \text{argmax}_{F\epsilon\Omega}(I,I') \qquad (3)$$

Where Ω is the set of the all possible solutions. Our objective is to maximize the quality measure (PSNR) calculated between the input image and the estimated one using the swarm intelligence algorithms.

4 Results and discussion

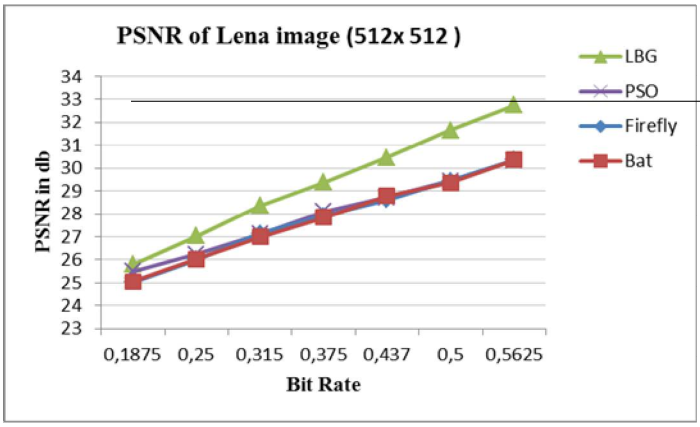
The LBG and the meta-heuristic algorithms proposed in this paper have been implemented in C language under Windows 7 operating system. The test machine consists of an Intel® Core™ i56-4590S, 3.00GHz processor with 8G of RAM. The experiments are carried out on the standard 512x512 test images (Lena, Barbara, Baboon, Cameraman, Pepper and GoldHill) . Due to space limitation, we report the results for only two images: Lena and pepper, **Fig. 1** and **Fig. 2**, respectively.

According to the PSNR values presented in **Fig. 3** and **Fig. 4**, the LBG algorithm shows a higher value than the meta-heuristic algorithms. This means that LBG produces a better codebook compared to the one generated by optimizing an objective function based on PSO, firefly and bat algorithms. For instance, the PSNR value using LBG algorithm (eg. Lena with book size = 256 / 512) results in a higher value (31,66 / 32,73) compared to the meta-heuristic based methods (PSO = 29,43 / 30,33), (FF= 29,44 / 30,35) and (Bat = 29,36 / 30,36).

Table 1. Reports the convergence time of different algorithms with different bit rate, it should be noted that the computation time using the LBG algorithm to generate the codebook is considerably lower than the computation time when using the swarm intelligence algorithms.

Fig.1. Lena image

Fig.2. Pepper image



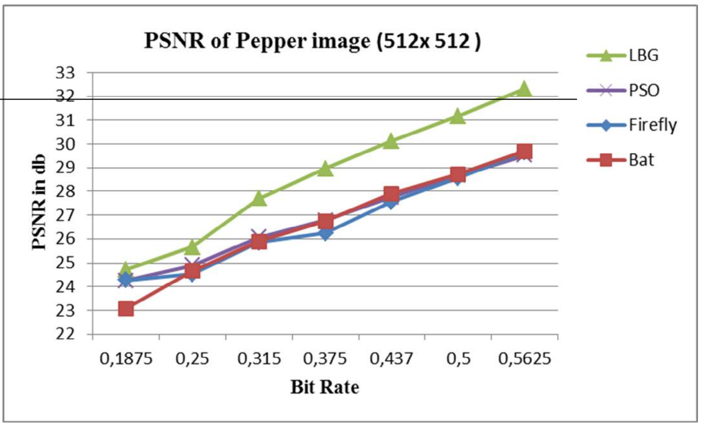


Fig. 3. The PSNR value of four vector quantization method for Lena image

Fig. 4. The PSNR value of four vector quantization method for pepper image

Table 1. Comparison of computation time between PSO, Firefly, Bat and LBG algorithms (size codebook M = 8 – 512)

	CPU time (second)						
	M=8	M=16	M=32	M=64	M=128	M=256	M=512
Lena							
LBG	0.05	0.21	0.57	1.07	1.78	3.66	6.31
PSO	3.201	7.133	14.462	28.408	56.191	110.682	222.425
FF	4.322	6.112	13.99	26.457	54.68	106.874	213.656
Bat	3.867	6.564	12.34	25.123	55.6	103.234	210.33
Pepper							
LBG	0.07	0.19	0.59	1.02	1.73	3.54	6.13
PSO	3.9	7.317	14.43	28.299	58.22	112.679	224.11
FF	4.865	7.626	16.5	30.324	54.097	106.903	211.502
Bat	4.561	6.987	15.1	29.321	52.067	101.127	209.098

5 Conclusion

Linde Buzo Gray (LBG) is the most popular method used to generate the VQ codebook, and the quality of the image compression depends on the choice of the codebook.

In this paper, a codebook generation based on meta-heuristic algorithms using a PSNR metric as an objective function is proposed. The numerical results presented in this paper indicate that the LBG algorithm is still the good choice to design the codebook compared to the use of swarm intelligence.

Knowing that the results obtained by the swarm intelligence algorithms depend on the choice of their parameters, our ongoing work is focusing on finding the most suitable parameters to overcome this problem. We may also make some changes in the original algorithms in order to further improve their performances.

References

[1] D. Salamon and G. Motta (2010). *Handbook of data compression. Fifth Edition*, Springer.

[2] Y. Linde, A. Buzo and R. M. Gray (1980). An algorithm for vector quantization design. *IEEE Transactions on Communications*, 28(1), 84-95.

[3] Gersho, A., & Gray, R. M. (2012). *Vector quantization and signal compression* (159). Springer

[4] Eberhart, R.C., Kenedy, J.: (1995). A New Optimizer Using Particle Swarm Theory.*In Micro Machine and Human Science 1995. MHS'95. Proceedings of the Sixth International Symposium*, 39-43. IEEE

[5] X.S. Yang (2010). *Nature-inspired metaheuristc algorithms*. Luniver Press

[6] X.S. Yang (2010). A new metaheuristic bat-inspired algorithm. *in NatureiInspired cooperative strategies for optimization (NISCO 2010) (65-74*). Springer Berlin, Heidelberg.

Multi-gene genetic programming for feature selection in DNA Microarrays

S.Sfaksi[1], A.Selmi[2], L.Djerou[3] and M.Moussaoui[4]

[1]. *LESIA Laboratory, University of Biskra, Algeria*
sara.sfaksi@gmail.com

[2]. *University of Biskra, Algeria*
aymenselmi681@gmail.com

[3]. *LESIA Laboratory, University of Biskra, Algeria*
ldjerou@yahoo.fr

[4]. *LESIA Laboratory, University of Biskra, Algeria*
moussaouimanel@yahoo.com

Keywords: Combinatorial optimization, DNA microarray, Feature selection, Metaheuristics, Multi-gene genetic programming.

1 Introduction

DNA microarray technology is currently experiencing an exceptional growth and generating tremendous interest in the scientific community due to its great potential to measure simultaneously the expression level of a great number of genes in tissue samples [1], which offers researchers the opportunity to obtain a comprehensive global view of genes regulatory network and then find out which ones are expressed in a specific tissue under different conditions [2]. Typically, DNA microarray dataset produces several thousand genes expression values for a small number of samples (usually less than 100), however, only a few genes from these high dimensional datasets are significant in classification task [3].

From the classification point of view, it is well known that, when the number of samples is much smaller than the number of features, classification methods may lead to data overfitting, meaning that one can easily find a decision function that correctly classifies the training data but this function may behave very poorly on the test data [1]. Moreover, data with a high number of features requires inevitably a large processing time. So, for analyzing microarray data, it is necessary to reduce the data dimensionality by selecting a subset of genes that are relevant for classification.

Computationally, genes selection problems can be expressed as a combinatorial optimization problem in which the search space involves a set of all possible subsets of variables among the set of *n* variables available (*n* is the large number of genes). This problem is known as an NP-hard problem because the selection of "good" subset of attributes requires potentially examining 2^n-1 possible subsets.

In this work, we investigate Multi-gene Genetic programming (MGGP) technique [6], which is a new subset of genetic programming (GP) that combines the ability of the standard GP in constructing the model structure with the capability of traditional regression in parameter estimation.

In MGGP, each individual (solution) is denoted as structure from 1 to ***Gmax*** genes represented as trees with a max depth ***Dmax***, that receives a set of input terminals x_j (j=1,…J) (expression values in our case) mapped via mathematical operators inserted in nodes to predict an output variable $\hat{\mathbf{y}}$ constituted by a weighted linear combination of the results of each of the trees/genes (*Gi* , i∈[1,Gmax]) in the multi-gene individual plus a bias term $\boldsymbol{d_0}$ [4]. The mathematical form of the multi-gene representation is shown as following: $\widehat{y} = \sum_{i=1}^{n} d_i \ G_i + d_o$ (1).

After creating an initial population of individuals, a process of some steps will be repeated until it reaches a specific criterion: calculate the fitness value of each individual, select the best among them as parents, reproduce new individuals through genetic operators (crossover, mutation and selection) and finally replace the weakest parents by the strongest ones. For that, MGGP requires to define certain parameters, which are: Iteration number, Gmax, Dmax, Crossover and mutation event probabilities and selection method, Nodes mathematical function (+, - , cos, sin …) and Fitness functions: maximize the performance and minimize complexity model (Minimal number of genes).

To develop the proposed MGGP-based model for feature selection in DNA Microarrays, an open-source software platform for symbolic data mining in MATLAB, namely GPTIPS 2 [5] is used in this work. For training data, we use two well-known public datasets described in (Table 1).

Dataset name	**Number of samples**	**Number of genes**	**Number of Classes**	**Description**
Prostate Cancer	102	12600	2 (50 Normal, 52 Tumor)	http://www.broad.mit.edu/cgi-bin/cancer/datasets.cgi
Lung Cancer	207	12600	2 (17 Normal, 190 Tumor)	http://www.broad.mit.edu/cgi-bin/cancer/datasets.cgi

Table 1. Microarray Datasets Description

The MGGP algorithm was run on the learning data and checked on the other subsets. The optimal model was selected considering its simplicity as well as its performance on the learning data. In this context, two criteria are optimized simultaneously; the complexity and the goodness-of-fit, by searching the so-called Pareto front (non-dominated solutions) set.

The complexity of the model is defined as the sum of nodes of all sub-trees within a tree. Minimizing this complexity is required to assess its simplicity. The goodness-of-fit is determined by the calculation of coefficient R^2 or the root-mean-square error (RMSE):

$$R^2 = \frac{\sum_{i=1}^{n} (\hat{y}_i - \bar{y})^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2} \qquad (2) \qquad\qquad \boldsymbol{RMSE} = \sqrt{\frac{\sum_{i=1}^{n} (\hat{y}_i - \bar{y})^2}{n}} \qquad (3)$$

To assess the performance of model, we minimize $E = 1 - R^2$ or minimize RMSE .
The parameter settings, used for the MGGP implementation, are shown in Table 2. These parameters were determined experimentally observing convergence of the objective functions over the generations.

Population size	**Number of generations**	**Gmax**	**Dmax**	**Training data**	**Test data**	**Crossover probability**	**Mutation probability**	**Nodes functions**	**tournament selection size**	**Pareto tournament**
					Prostate Cancer					
500	200	8	6	90%	10%	0.8	0.1	+ , -	5	0.3
					Lung Cancer					
500	200	8	7	90%	20%	0.8	0.1	+ , -	5	0.3

Table 2. MGGP parameter settings

After 200 iterations on each dataset, two prediction models were selected as the best with 49 relevant genes for prostate dataset and 50 for Lung dataset (Table 3):

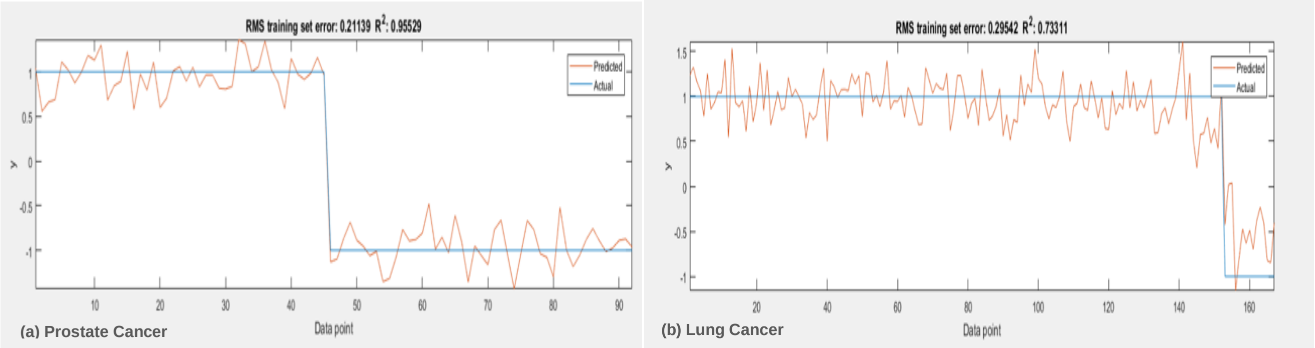
	The selected genes
PROSTATE CANCER **(49 genes selected)**	'768_at' - '40349_at' - '36569_at' - '38665_at' - '39845_at' - '40993_r_at' - '40025_at' - '37093_at' - '36802_at' - '2073_s_at' - '36629_at' - '39430_at' - '38976_at' - '725_i_at' - '672_at' - '35278_at' - '340_at' - '39423_f_at' - '38876_at' - '38090_at' - '36686_at' -'36043_at' - '1260_s_at' - '39720_g_at' - '769_s_at' - '32545_r_at' - '40536_f_at' - '32436_at' - '35429_at' - '33881_at' - '36893_at' -'41252_s_at' - '40336_at' - '38287_at' - '39840_at' - '39562_at' - '34185_at' - '32598_at' - '35048_at' - '41376_i_at' - '38764_at' -'34137_at' - '34551_at' - '40233_at' - '37639_at' - '39557_at' - '40433_at' - '876_at'
LUNG CANCER **(50 genes selected)**	*'33176_at'- '37604_at'- '39045_at'- '36535_at'- '619_s_at'- '37997_r_at'- '41776_at'-'41695_at'-'32947_at'- '35730_at' - '36447_at' -'33910_at'- '37464_at'- '32082_at'-'37132_at'-'37689_s_at'-'40092_at'- '40387_at'- '1857_at'- '1175_s_at'- '41087_at'- '35019_at'- '34829_at'-'585_at'-'41464_at'-'37094_at'- '32242_at'- '31859_at'-'36734_at'- '35705_at'- '38308_g_at'- '31329_at'- '33966_at'-'40275_at'- '35990_at'- '31506_s_at'- '38856_at'- '32244_at'- '34003_at'- '35042_at'- '444_g_at'-'38508_s_at'- '41382_at'-'34640_at'- '35363_at'- '1224_at'- '36515_at'- '40868_at'-'31513_at'-'35352_at'*

Table 3. The selected genes in best prediction models

Prediction accuracy metrics values calculated for each dataset model (Table 4) show the efficiency of our method, in which, whenever R^2 is maximized and RMSE is minimized, we get a good correlation between real and predicted values as it is shown in figure 2:

	EVALUATION METRICS					
	R^2 : Correlation Coefficient	RMSE: Root Mean-Square Error	MSE: Mean Square Error	SSE: Sum of Square Error	MAE : Mean Absolute Error	MAXE : Maximum Error
PROSTATE CANCER	0.9553	0.2114	0.0447	4.1111	0.1639	0.5225
LUNG CANCER	0.9264	0.1537	0.0236	2.9591	0.1133	0.5563

Table 4. Prediction accuracy metrics values



(a) Prostate Cancer

(b) Lung Cancer

Figure 2. Real and predicted values correlation

In order to prove the quality of the optimal selected subsets, three classification methods were used. Classification rates values presented in table 5 show the efficiency of using these subsets (ii) than using all genes (i), whatever the induction method used.

	SVM		Symbolic regression		Decision tree	
	i	ii	i	ii	i	ii
Prostate cancer	94.1176%	97.0297%	93.1373%	97.0297%	83.3333%	89.1089%
Lung cancer	98.032%	99.0291%	89.8551%	99.5146%	97.1014%	96.1165%

Table 5. Classification rates values

References

[1] Huerta, E.B., B. Duval, and J.-K. Hao. *A Hybrid GA/SVM Approach for Gene Selection and Classification of Microarray Data*. 2006. Berlin, Heidelberg: Springer Berlin Heidelberg.
[2] Dashtban, M., M. Balafar, and P. Suravajhala, *Gene selection for tumor classification using a novel bio-inspired multi-objective approach.* Genomics, 2018. 110(1): p. 10-17.
[3] Jain, I., V.K. Jain, and R. Jain, *Correlation feature selection based improved-Binary Particle Swarm Optimization for gene selection and cancer classification.* Applied Soft Computing, 2018. 62: p. 203-215.
[4] Orove, J.O., Osegi, N.E., and Eke, B.O. *A Multi-Gene Genetic Programming Application for Predicting Students Failure at School*. 2014. *CoRR, abs/1503.03211. D.P.*
[5] Searson DP, GPTIPS 2: an open-source software platform for symbolic data mining, 2014 (arXiv preprint arXiv:14124690).
[6] D.P. Searson, D.E. Leahy, M.J. Willis, *GPTIPS: an open source genetic programming toolbox for multigene symbolic regression*, Proceedings of the International Multiconference of Engineers and Computer Scientists: Citeseer 2010, pp. 77–80.

A modified cuckoo search algorithm for unsupervised satellite image classification

Kaouter Labed[1], Hadria Fizazi[2], Habib Mahi[3]

1. Ecole Normale Supérieure d'Oran, Oran, Algérie
klabed@gmail.com

2. Université Mohamed Boudiaf – USTOMB, Oran,Algérie
fizazi@univ-usto.dz
3. Division Observation de la Terre, Centre des Techniques Spatiale, Arzew, Oran, Algérie
hmahi@cts.asal.dz, mahihabib@yahoo.fr

Keywords : Remote sensing cuckoo search, clustering, modified cuckoo search.

1 Introduction

For remote sensing applications, classification is an important task where the pixels in the images are classified into homogeneous regions; each one of them corresponds to some particular landcover type [1]. In classification image, a distinction is often made between supervised and unsupervised methods. Training samples are required by supervised algorithms to perform the classification. It is frequently the analyst who gives these data. By contrast, the unsupervised classification (clustering) needs fewer interactions with the analyst; the most necessary one is the image regions number which requires in many cases multiple runs for their selection [2]. A number of different techniques has been already suggested to have regions number automatically but until now no global method is adopted and the choice of regions number by multiple run is the mostly used [2].

Clustering is defined as a process based on specific characteristics of a data set and oriented to partition this data set into a proper number of groups [3]. In other words, the aim of clustering is to find the optimal partition of a specified n data points into c subgroups, such that the inter-class distance is as far as possible whereas the intra-class distance is as close as possible [4]. many clustering algorithms have widely been used to solve remote sensing images classification such as K-means, Fuzzy C-means (FCM), ISODATA [4]. To achieve their goal, clustering methods try to optimize an evaluation function, and use cluster validation indexes for evaluate the quality of classification [5]. Some hybrid image segmentation methods combine two or more above approaches and some others methods used multiobjective algorithms [6] [7].

Biological systems have natural capability to adapt to changes by learning, it's evolving, resiliant and robust. Bio-inspired algorithms (BIAs) are inspired by the behavior observed in these biological systems, and are increasingly used in solving problems in virtually any area in between image processing [8]. There are a lot of branches of BIAs, the most known examples are swarm intelligence (SI), evolutionary algorithms (EAs), artificial neural networks (ANNs) and bacterial foraging algorithms (BFAs) [9]. In general, methods in BIAs used collective intelligence which is a highlevel phenomenon that emerges naturally from the interplay of collaboration and competition of many individuals of a population. It is usually defined as the ability of a group to solve problems than its individual members cannot solve on their own [10]. BIAs has drawn considerable research interests in the area of image processing. In [8] the authors present a survey of scientific paper related to BIAs applied in image processing and specially for image segmentation; feature extraction; image enhancement and image registration. Hence, the authors in [8] concluded that BIAs Applications were highly growing in number between 1995 and 2010. Also from 130 papers on BIAs algorithm studied by [8], 80 papers were applied in image processing. Most of the researches were centered in the area of image extraction (47%) and image segmentation (31%). Only a small number of researches were done in image enhancement (13%) and registration (9%).

BIAs algorithms are usually considered as clustering based methods. In the literature, we find an intensive research by scientists using different BIAs algorithms as, genetic algorithms, ant colony optimization, artificial immune system, artificial bee colony, and particle swarm optimization [8][11]. We find also, efficient hybrid optimization algorithms and multiobjective algorithms [6].

Cuckoo Search is a bio-inspired meta heuristic algorithm that simulates the aggressive reproduction strategy of some cuckoo species [9]. This method is very used in solving optimization problems and gives good results in remote sensing images classification [12].

In this paper, a new version of cuckoo search algorithm using kernel distance and fuzzy logic is proposed for unsupervised satellite image classification. The fuzziness logic will be employed to overcome the problem of incertitude and imprecision. The use of Gaussian radial basis function (GRBF) aims to enhance robustness against noise and ability to classify complicated data structure .

The proposed CS algorithm is adopted to find K cluster centers from the data set by optimizing a fitness function. Hence, each cuckoo i (nest, solution) is represented by a vector Kxl, where l represents the feature space dimension of each center. As an example we can take l=3 when using radiometric values of a pixel (red, green, blue).Thus, mathematically speaking, i={Ci1, Ci2,...,CiK} where CiK is the center of the Kth cluster. Concerning the Fitness function, a panoply of cluster validity indices exist in literature and are used as objectives functions for bio-inspired algorithms in image segmentation. In this work, the well known Jm index is considered as a fitness function. This choice is justified by the fuzziness nature of this index, and the good results obtained with other bio-inspired algorithms [13] [1]. The Jm index is defined as follows [14]:

$$J_m = \sum_{i=1}^{N}\sum_{j=1}^{K} u_{ij}^{m}\left\|x_i - c_j\right\|^2$$

The membership values uij, are computed as follows [15]:

$$u_{ij} = \frac{1}{\sum_{l=1}^{K}\left(\frac{\left\|x_i, c_j\right\|}{\left\|x_i, c_l\right\|}\right)^{\frac{2}{m-1}}}$$

Where:

m: real number greater or equal than 1(1<=m<∞)

uij: degree of membership of a pixel xi in the cluster j.

N: number of items to be clustered in the data set.

K: number of clusters.

cj: jth cluster center.

xi: ith data set item.

The kernel mapping on the Jm index using the Gaussian function is introduced in the purpose is to calculate the clustering quality after mapping the data into a high dimensional space:

$$J_m^{\varphi} = 2\sum_{i=1}^{N}\sum_{j=1}^{K} u_{ij}^{\varphi}\left(1 - k\left(\overrightarrow{z_i} - \overrightarrow{c_j}\right)\right)$$

Comparisons with standard CS and other bio-inspired algorithms were performed on different data sets. Experimental results confirm the effectiveness of the proposed approach.

References

[1] Bandyopadhyay, S., U. Maulik, and A. Mukhopadhyay. (2007). Multiobjective genetic clustering for pixel classification in remote sensing imagery. Geoscience and Remote Sensing 45 (5):1506–11.

[2] Ming-Der, Y. (2007). A genetic algorithm (GA) based automated classifier for remote sensing imagery. Journal Canadien De Télédétection 33 (3):203–13.

[3] Girolamo, F. & Antonio, G. (2013). An unsupervised multi-swarm clustering technique for image segmentation, Swarm and Evolutionary Computation, 11, 31-45.

[4] Swagatam, D., and K. Amit. (2009). Automatic image pixel clustering with an improved differential evolution. Applied Soft Computing 9 (1):226–36.

[5] Bensaid AM, Hall LO, Bezdek JC, Clarke LP, Silbiger ML, Arrington JA and Murtagh RF, (1996).: Validity-guided (Re) Clustering with applications to image segmentation, IEEE Transactions on Fuzzy Systems, 4:112-123.

[6] Bong C and Rajeswari M, (2011).: Multi-objective nature-inspired clustering and classification techniques for image segmentation, Applied Soft Computing 11, 3271–3282

[7] Chin-Wei.B et Rajeswari. M, (2010): Multiobjective Optimization Approaches in Image Segmentation – The Directions and Challenges, Int. J. Advance. Soft Comput. Appl., Vol. 2, No. 1.

[8] Abdul Khalid NE, Ariff N Md, Yahya S, and Noor NM, (2011): A Review of Bio-inspired Algorithms as Image Processing Techniques, In proceeding of: Software Engineering and Computer Systems - Second International Conference, ICSECS 2011, Kuantan, Pahang, Malaysia, June 27-29.

[9] Yang X-S, (2010): Nature-Inspired Metaheuristic Alogorithms, Second Edition, Luniver Press.

[10] Dariusz K and Heitor SL, (2012): Nature-inspired collective intelligence in theory and practice, Editorial, Information Sciences 182.

[11] Jose Alfredo F Costa and Jackson G de Souza, (2011): Image Segmentation through Clustering Based on Natural Computing Techniques, In: Image Segmentation, Pei-Gee Ho (Ed.), pp. 57-82, ISBN: 978-953-307-228-9, InTech. Austria.

[12] Kaouter Labed, Hadria Fizazi, Habib Mahi & Inés M. Galvan (2018) A Comparative Study of Classical Clustering Method and Cuckoo Search Approach for Satellite Image Clustering: Application to Water Body Extraction, Applied Artificial Intelligence, 32:1, 96-118,

[13] Fizazi H., Beghoura M.A., An Enhanced Bio-Inspired Firefly Algorithm for Remote Sensing Images, in: proc Colloque sur l'optimisation et les Systèmes d'information, COSI'12, Tlemcen, 2012, pp. 337-348.

[14] J.C.Bezdeck, FCM: Fuzzy C-Means algorithm, Computers and Geoscience, Vol 10, pp.191-203, 1984.

[15] J. Zexuan, X. Yong, C. Qiang, S. Quansen, X. Deshen and D.F. David, “Fuzzy c-means clustering with weighted image patch for image segmentation, ” Applied Soft Computing. Vol 12, pp.1659–1667, 2012.

Embedded System for Template Matching using Swarm Intelligence

Alexandre de V. Cardoso[1,2], Nadia Nedjah[2], and Luiza de Macedo Mourelle[2]

[1] Brazilian Navy Weapons Systems Directorate, Brazilian Navy,
Rio de Janeiro, Brazil
alexvcardoso@yahoo.com.br

[2] Postgraduate Program in Electronics Engineering, State University of Rio de Janeiro,
Rio de Janeiro, Brazil
nadia|ldmm@eng.uerj.br

1 Introduction

Image and video processing led to advances in important research areas. New sensors and intelligent equipments capable of capturing, storing, editing and transmitting images accelerate decision processes, enabling more robust strategies. The time to obtain a true information and process it is directly responsible for the success of these operations. A slow search process may delay decision taken until recorded data become insufficient or even inefficient at that moment [6]. An airplane pilot observes the environment to avoid any possible collision. Similarly, a plane that uses an automatic guidance system acquires environment information to decide path changes due to random objects in its way [11]. Target recognition and tracking using image sequences can be used to provide solutions for surveillance and monitoring systems, guidance [4], remote biometrics authentication [3], fire control, among many other applications.

Template matching (TM) is one of the most used techniques for finding patterns in images [2, 8]. It consists basically of finding a small image, termed as the template, inside a larger image. Among the methods used to evaluate the matching process, Pearson's Correlation Coefficient (PCC) is widely used because of its invariance to global linear brightening changes [10], but this task is computationally very expensive, especially when using large templates with an extensive image set [13]. In order to find the maximum correlation point between the image and the template, we used Artificial Bee Colony (ABC), one of the swarm intelligence strategies. ABC is an optimized search strategy [1], inspired by the bees foraging behaviour.

This work proposes an implementation of the template matching by an embedded system through a codesign methodology, running ABC search optimization in software, while the required computation of PCC in hardware via a dedicated co-processor. In order to evaluate the proposed design, the processing time obtained by the software-only implementation and that obtained by the codesign system are compared.

The rest of this paper is organized into five sections. First, in Section 2, we present the template matching, correlation and ABC concepts as they are used in this work. Then, In Section 3, we describe the proposed hardware. In Section 4, we present the results obtained with and without the coprocessor. Finally, in Section 5, we draw some conclusions and point out some new directions for future work.

2 Basic Concepts

Template Matching is used in image processing to determine the degree of similarity between two images of the same size. In the present work, TM compares a pre-defined pattern, called template, to pieces of a bigger image in order to recognize its incidence. The similarity evaluation considers all the possible pixels of bigger image, then the set of pixels, that provide the highest correlation degree, is identified as the location of the pattern inside the image. Pearson's Correlation Coefficient, a dimensionless number that means the normalized cross correlation, measures the similarity between these images and it can be computed as defined by equation 1:

$$PCC(A,P) = \frac{\sum_{i=1}^{N}(p_i - \overline{p}) \times (a_i - \overline{a})}{\sqrt{\sum_{i=1}^{N}(p_i - \overline{p})^2 \times \sum_{i=1}^{N}(a_i - \overline{a})^2}}, \tag{1}$$

wherein p_i is the intensity of the pixel i in the template P; $\overline{p}$ is the average intensity of the pixels of the template; a_i is the intensity of the pixel i in the patch A from the analyzed image; $\overline{a}$ is the average intensity of the pixels in the patch of the image A. Note that template P and the patch A of the image must have the same dimensions. The PCC always assumes real values in $[-1, +1]$. Coefficient $+1$ means a perfect positive correlation of the compared variables while coefficient -1 means a perfect negative correlation of the compared variables.

In ABC, the food sources in the search space represent possible solutions of the problem and the amount of nectar available is associated to the quality of the solution, defined by the objective function. This strategy mimics activities of communication, task allocation, swarm placement choice, breeding and reproduction, search for nectar and feromine diffusion. Three types of bees are defined: employed, onlooker and scouts. The employed bees are responsible for the routes of the food sources and inform other bees about the quality of the source food. The onlookers wait, in the hive, for new information from the employed ones. The scouts search for new food sources, regardless from others already known. The number of solutions is the quantity of scouts and onlookers together.

An artificial employed bee chooses a food source according to the probability P_i, related to the source, defined by equation 2:

$$P_i = \frac{fit_i}{\sum_{i}^{SN} fit_n}, \tag{2}$$

wherein fit represents the amount of nectar of the i-th source and SN is the amount of food sources, which corresponds to the total number of bees. A new neighbouring source is defined, based on an old one, using equation 3:

$$v_{ij} = x_{ij} + \phi_{ij} \times (x_{ij} + x_{kj}), \tag{3}$$

wherein k and j are chosen so that $k \in [1, 2, ..., SN]$ and $j \in [1, 2, ..., D]$, where D is the number of dimensions of the search space. Position k is chosen randomly, but different from i. Term ϕ_{ij} is a random number in the interval $[-1, 1]$, that controls the choise of sources nearby x_{ij} and represents the visual comparison done by a bee between two sources close to each other. If x_{ij} and x_{kj} are different, x_{ij} will vary according to the magnitude of the difference. When it is close to the optimum solution, the size of this step tends to be small.

The food sources discarded by the bees are sustituted for new ones through the employed bees. In the ABC strategy, a source will be abandoned if it does not have its quality improved by its neighbour during a predefined number of cicles. For a source x_i and $j \in [1, 2, ..., D]$, the employed bee finds a new food source according to equation 4:

$$x_i^j = x_{min}^j + rand(0,1) \times (x_{max}^j + x_{min}^j). \tag{4}$$

After each candidate position v_{ij} for source is proposed and visited by a bee, its performance is compared with the old one. If better then it is used and the other one forgoten, otherwise the old one remains in memory. This means a greedy mecanism for the selection operator between sources [1].

The ABC strategy, presented in algorithm 1, uses the number of food sources (SN), which is equal to the number of bees, either employers or onlookers, the limit value of the search space (D) and the maximum number of cycles (MNC). In the algorithm, initially, the food sources are visited by the employed bees and evaluated through the objective function. If a new food source is found nearby with a better value, it replaces the old one, otherwise the new one is rejected. Employed bees share this information with onlooker bees and these choose the source according to the probability of finding food there, based on the information provided by the employed bees.

The onlooker bees are also capable of finding new food sources near the one visited and adopting it, if it is better than the previous one, otherwise it is rejected.

Algorithm 1 ABC Search Algorithm

```
Initiate the food sources $x_{ij}$, $i = 1 ... SN$ and $j = 1 ... D$
while $CycleNumber < MNC$ do
    Compute $v_{ij}$
    Evaluate the new solutions by objective function
    Apply the greedy selection process
    Compute $P_i$
    Compute $v_{ij}$, depending on $P_i$ and evaluate them by the objetive function
    Apply the greedy selection process
    Find the solution rejected by the employed bee
    Substitute the rejected source, if there is one, by a new source $x_{ij}$ randomly generated by
    Equation 4
    Store the best solution (source) found so far
end while
Return best solution
```

3 Codesign Architecture

The codesign approach is a methodology to develop an integrated system using hardware and software components, to satisfy performance requirements and cost constraints [9]. The final target architecture often has software components executed by a soft processor that is aided by some dedicated hardware components developed especially for the application. PCC is the most expensive computation of the template matching process and the choice for hardware implementation, as a coprocessor. The inherent parallelism of the hardware gives an advantage to improve the processing time, allowing a real-time execution. The search process for the location with the maximum correlation degree is implemented by ABC, executed as a software by a general purpose processor.

Figure 1 presents the proposed architecture for the coprocessor responsible for the correlation computation for two given images, as defined in equation 1. The architecture was designed to operate in pipeline and is composed of three blocks, which correspond to each one of the three pipeline stages. During the rising edge of the clock signal, the coprocessor receives three data: `data_p`, which is a 1-byte pixel from the template; `data_ac`, which is a 1-byte pixel from the image to be compared; and `data_am`, which is a 1-byte pixel from the next image.

All the images applied to the comparison process are 64x64 pixels, consisting of a total of 4096 pixels. The computations are performed 1 block at a time and the obtained results are passed on to the next block at every synchronism pulse. This pulse is generated by component `synchro` at every 4103 clock cycles. The coprocessor provides the value of the PCC (`result`), as 32 bits two's complement; a flag, indicating the end of operation (`flag_end`); an error signal (`error`), showing whether the result is valid, in case of a division by zero.

Block 1 represents the pipeline first stage and is responsible for computing the pixel average of the images to be compared. Block 2 represents the pipeline second stage and is responsible for computing the three sums of equation 1. This block consists of two components `subt_A2`, three components `mult_CLK` and three components `sum_A2`. Component `subt_A2` performs the subtraction in two's complement of the image pixel's intensity of the averages from Block 1. Component `mult_CLK` performs, during one clock period, the multiplication of the results provided by components `subt_C2`. Component `sum_C2` executes the sum of the products provided by components `mult_CLK`, in two's complement.

Block 3 in figure 1 introduces the pipeline third stage, and implements the main multiplication, the square-root and the division of equation 1. It is composed by: component `mult_CLK`, which performs the multiplication of the denominator in equation 1 in one clock period; component `SQRT` to calculate the square-root operation; and component `div_frac_C2`, which performs the division result with 2^{-24} precision. The latter provides the output signals of the coprocessor. The operation of this block is controlled by a finite state machine (FSM), which coordinates the correct sequence operation of the block's components. A very effective method to calculate the square root was developed by ancient Babylonians [7]. Based on this, called *Babylonian*, the component SQRT in Block 3 has been implemented, in hardware, using an iterative process of N steps, that is set according to the required result precision. Component div_frac_C2 in Block 3 implements the division calculus in hardware to obtain the required result.

The three blocks have output registers that are loaded only when the stage task is completed. At a synchronism pulse, the components subt_C2, mult_CLK and subt_C2 are reset and the FSM returns to its initial state.

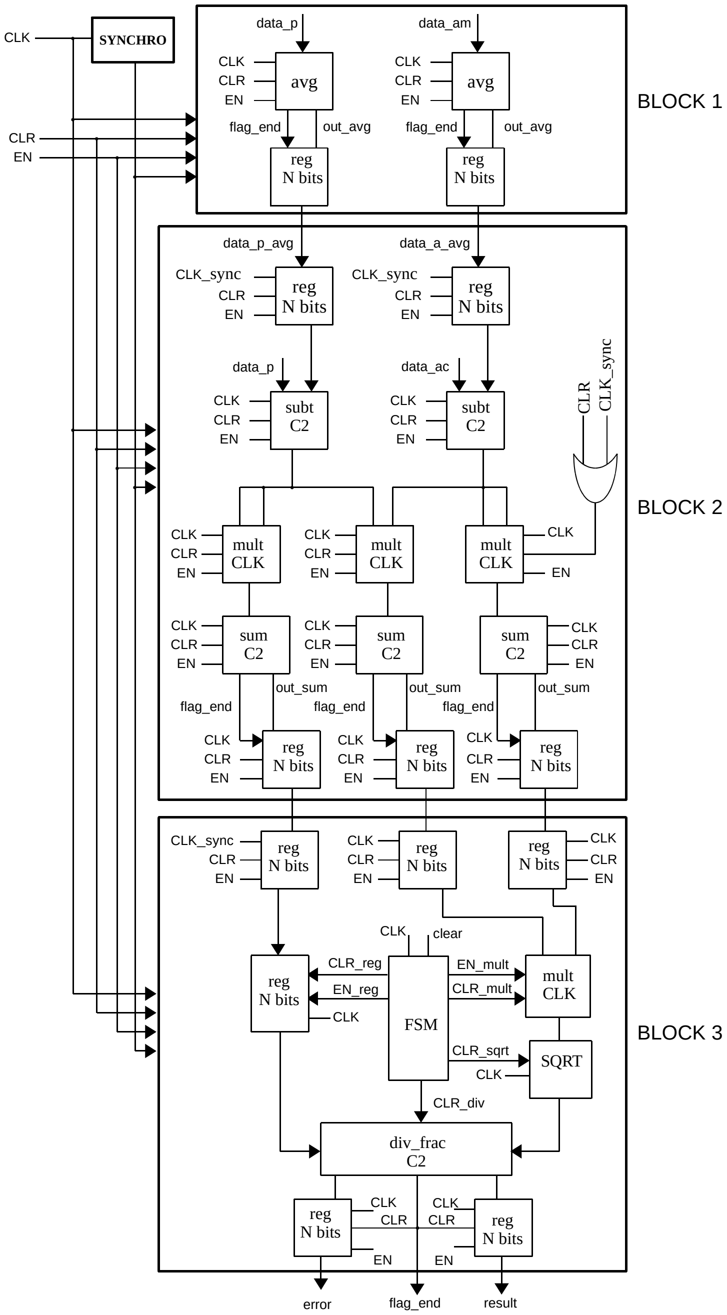
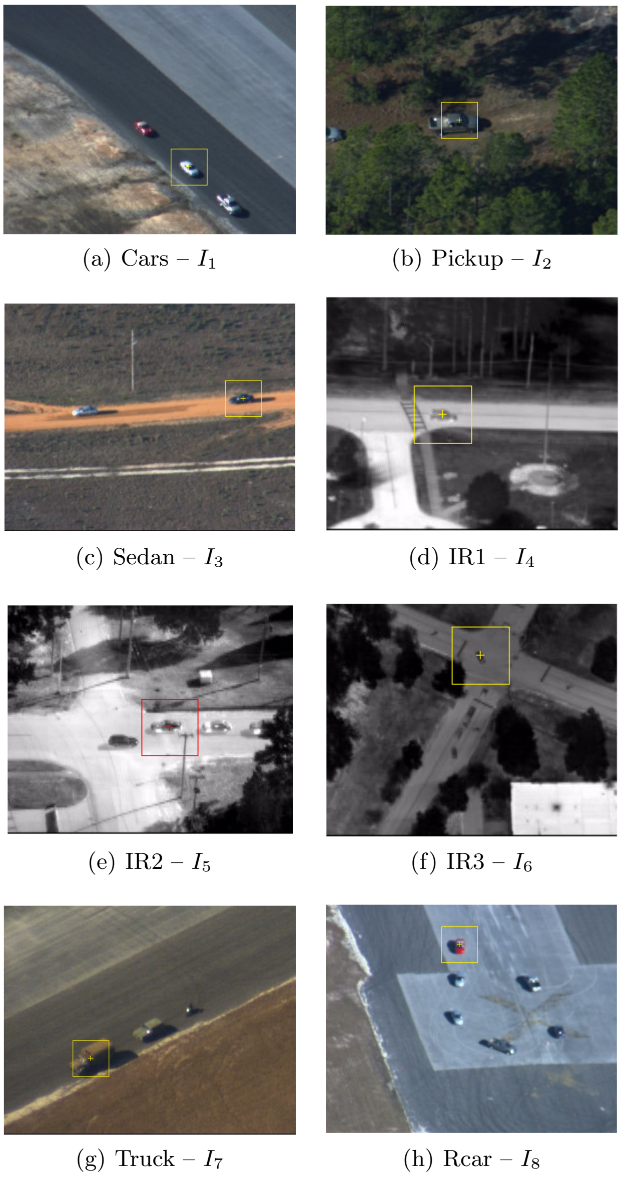


Fig. 1. Coprocessor architecture

4 Performance Results

Figure 2 shows the images used in this work [5] and the corresponding templates, highlighted by the inner square. All experiments are for 1000 template searches performed on each image. The results are for average processing time and success rate. The correlation for each pixel is calculated extracting a patch, of the same size as the template, and computing the corresponding PCC at its center. It is noteworthy to point out that the limits of the main image are completed with zeros. The objective is to find templates using TM in less than 33ms, enough time to keep a tracking rate for videos of 30fps, with success rate better than 95%.



(a) Cars – I_1

(b) Pickup – I_2

(c) Sedan – I_3

(d) IR1 – I_4

(e) IR2 – I_5

(f) IR3 – I_6

(g) Truck – I_7

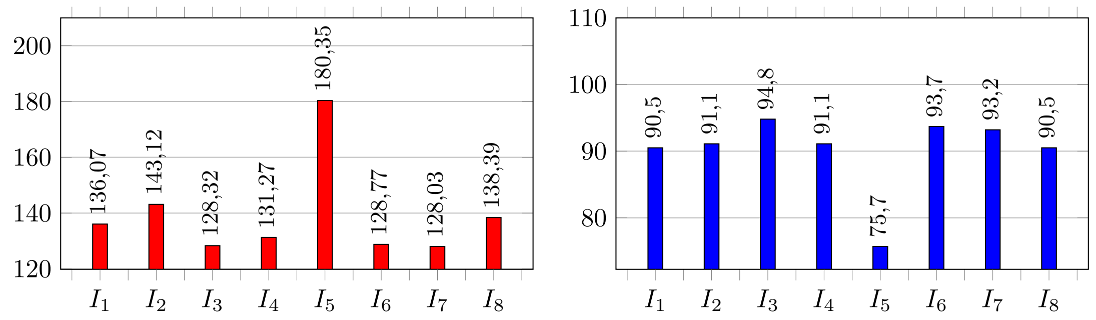
(h) Rcar – I_8

Fig. 2. Reference images used in the tests

The aim is to simulate video tracking, in which the frame sequence during the process receives the position of target found on the last frame and starts the new search around it. Besides that, the adjustment of the window size proportionally to the target's speed helps to enhance the success rate and the processing time during tracking, because a small variation of the target position in two consecutive frames allows using a smaller search window.

The proposed system was evaluated using three scenarios, while keeping the ABC running by the embedded processor: PCC_{SO}, in which the PCC is also executed by the embedded processor; PCC_{SH}, in which the PCC is calculated by the coprocessor in serial mode; PCC_{PH}, in which the PCC is calculated by the coprocessor in pipeline mode. The synthesis was implemented using the *Smart Vision Development Kit* rev 1.2 (SVDK) of *Sensor to Image* [12] and Xilinx PicoZed 7Z015 System-On-Module rev. C. It required 11% of flip-flops, 39% of LUTs, 25% of buffers and 69% of block RAMs. Due to synthesis time constraints, the coprocessor ran at 25MHz.

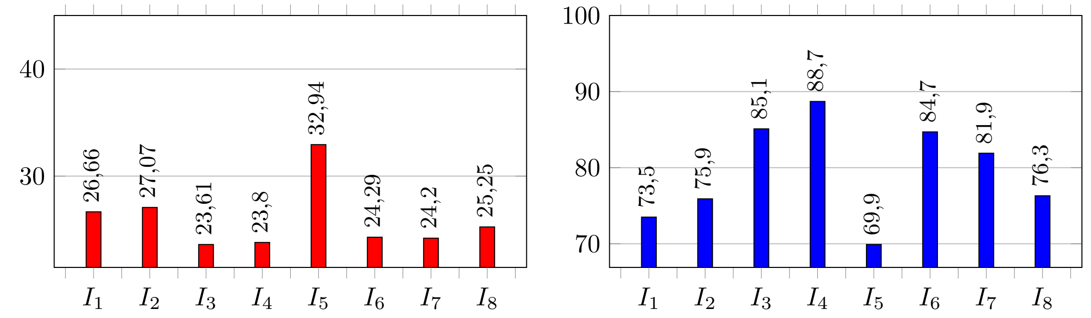
The ABC algorithm was configured with eight bees, which correspond to the number of food sources, and the number of iterations MNC with 1, which corresponds to the exhaustion of on source. Figure 3 shows the results yield by the system without the coprocessor. Figure 4 shows the results yield by the system with the coprocessor in serial mode. Figure 5 shows the results of the system with the coprocessor in pipeline mode, using a window of 101 x 101 *pixels*, while figure 6 of the same system using a window of 51 x 51 *pixels*.



(a) Processing time (ms)

(b) Success rate (%)

Fig. 3. Results for ABC without the coprocessor

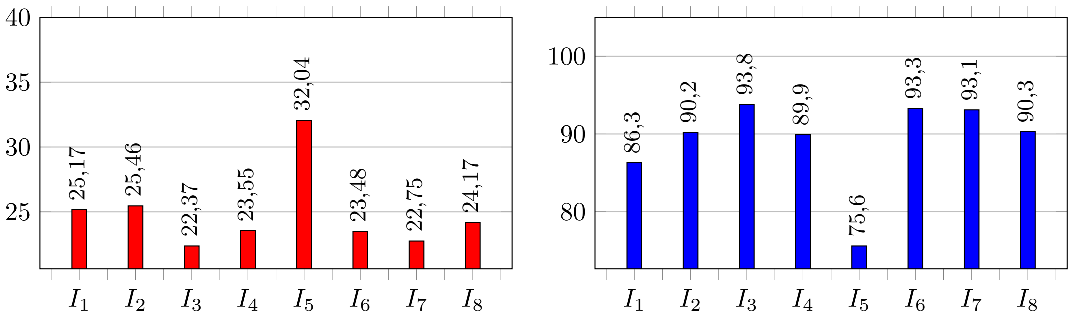


(a) Processing time (ms)

(b) Success rate (%)

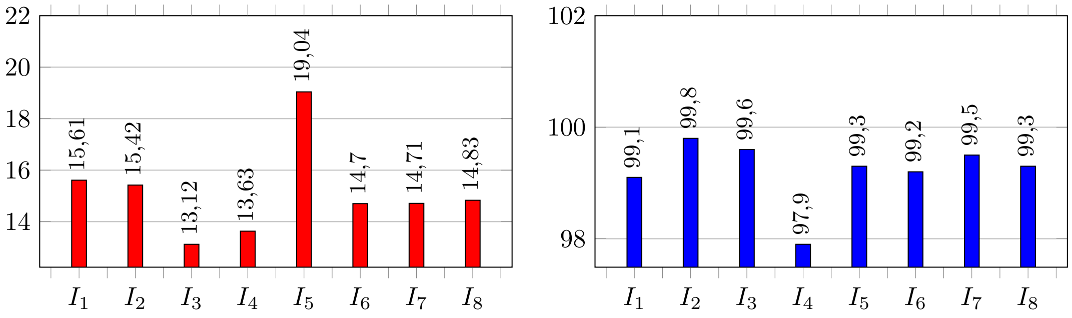
Fig. 4. Results for ABC with the coprocessor in serial mode

The results obtained were compared to those previously obtained using PSO as the optimization search strategy [14]. Concerning the software only implementation, figure 7 shows that any of the strategies achieved the required processing time (33ms). Figure 8 shows that only PSO achieved a good success rate and the others only in some cases, but no better than PSO.



(a) Processing time (ms) (b) Success rate (%)

Fig. 5. Results for ABC with the coprocessor in pipeline mode, using a window of 101 x 101 *pixels*



(a) Processing time (ms) (b) Success rate (%)

Fig. 6. Results for ABC with the coprocessor in pipeline mode, using a window of 51 x 51 *pixels*

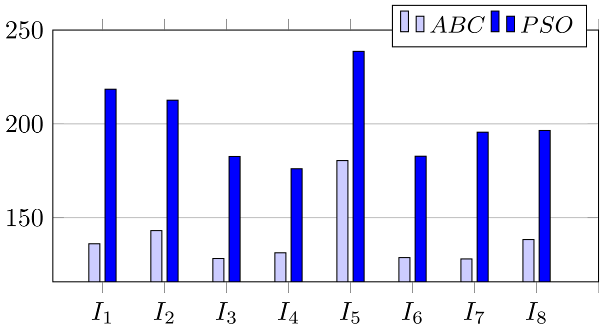


Fig. 7. Processing time without coprocessor (ms)

Concerning the use of the coprocessor in serial mode, figure 9 shows that ABC achieved the required processing time, but PSO only in some cases. Figure 10 shows that PSO achieved the required success rate in most of the cases, but ABC did not in all cases.

Figure 11 shows that both strategies achieved the required time, while PSO yield smaller times than ABC. Figure 12 shows that PSO yield better success rate in all cases, while ABC did not performe well in a specific case (I_5).

5 Conclusion

In this work, we proposed an implementation of the template matching by an embedded system through a codesign methodology, running ABC search optimization in software, while the required

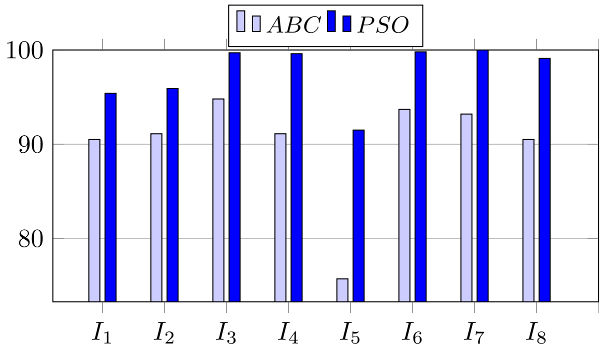


Fig. 8. Success rate without the coprocessor (%)

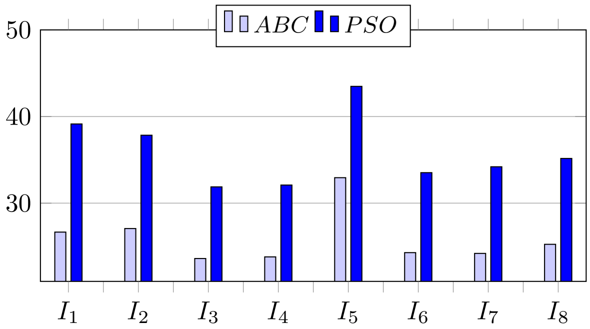


Fig. 9. Processing time with coprocessor in serial mode (ms)

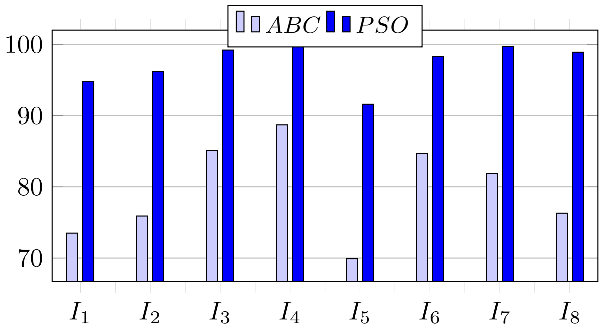


Fig. 10. Success rate with coprocessor in serial mode (%)

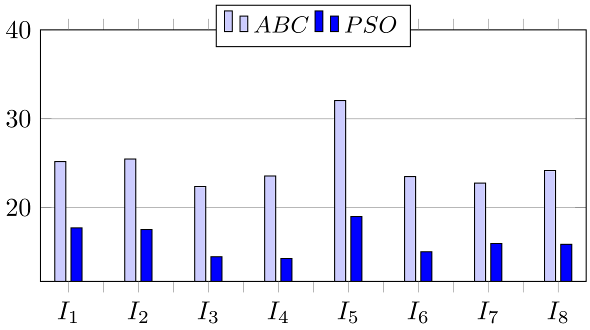


Fig. 11. Processing time with coprocessor in pipeline mode and a window of 101 x 101 *pixels* (ms)

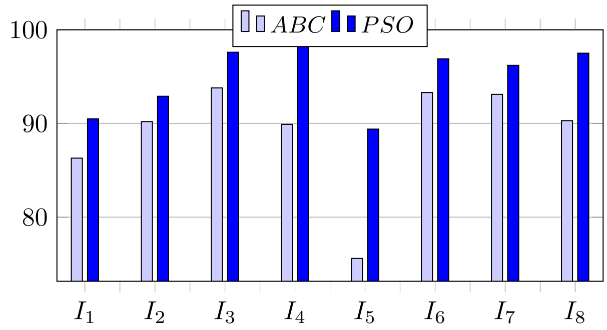


Fig. 12. Success rate with coprocessor in pipeline mode and a window of 101 x 101 *pixels* (%)

computation of PCC implemented in hardware via a dedicated coprocessor. In order to evaluate the proposed design, the processing time obtained by the software-only implementation and that obtained by the codesign system were compared to previous results obtained when using PSO as the search optimization. The proposed codesign system showed to be capable of achieving real-time execution for object tracking.

The results showed that the sofware only implementation is not viable for object tracking, yielding results greater than 33ms. On the other hand, the introduction of the coprocessor improved the performance of the system, yielding processing times smaller than 33ms. When using the coprocessor in serial mode, ABC yield smaller processing times than PSO, while when using the coprocessor in pipeline mode, PSO yield smaller processing times than ABC. It is our intention to investigate the behaviour of the codesign system in the presence of other swarm intelligence techiques, such as Bacterial Foraging Optimization Algorithm (BFOA), Firefly Algorithm (FA), Fireworks Algorithm for Optimization (FWA).

References

1. Artificial bee colony (abc) optimization algorithm for solving constrained optimization problems. In *International Fuzzy Systems Assocde Association World Congress*, pages 789–798, Cancun, Mexico, 2007. Springer.
2. Kavita Ahuja and Preeti Tuli. Object recognition by template matching using correlations and phase angle method. *International Journal of Advanced Research in Computer and Communication Engineering*, 2(3):1368–1373, 2013.
3. Ben Benfold and Ian Reid. Stable multi-target tracking in real-time surveillance video. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3457–3464. IEEE, 2011.
4. Hyunjin Choi and Youdan Kim. Uav guidance using a monocular-vision sensor for aerial target tracking. *Control Engineering Practice*, 22:10–19, 2014.
5. Robert Collins, Xuhui Zhou, and Seng Keat Teh. An open source tracking testbed and evaluation web site. In *IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, volume 35, 2005.
6. Cheng Jihang, Dong Xinwen, Zheng Nan, and Luo Tianwen. Study on image target interpretation decision-making based on analytic hierarchy process and clustering analysis. In *Computer Science and Network Technology (ICCSNT), 2012 2nd International Conference on*, pages 2045–2049. IEEE, 2012.
7. Olga Kosheleva. Babylonian method of computing the square root: Justifications based on fuzzy techniques and on computational complexity. In *Fuzzy Information Processing Society, NAFIPS*, pages 1–6, USA, 2009. IEEE.
8. T Mahalakshmi, R Muthaiah, P Swaminathan, and T Nadu. Review article: an overview of template matching technique in image processing. *Research Journal of Applied Sciences, Engineering and Technology*, 4(24):5469–5473, 2012.
9. Nadia Nedjah and Luiza de Macedo Mourelle. *Co-design for system acceleration: a quantitative approach.* Springer Science & Business Media, 2007.
10. Nazil Perveen, Darshan Kumar, and Ishan Bhardwaj. An overview on template matching methodologies and its applications. *IJRCCT*, 2(10):988–995, 2013.
11. David Salmond. Tracking and guidance with intermittent obscuration and association uncertainty. In *Information Fusion (FUSION), 2013 16th International Conference on*, pages 691–698. IEEE, 2013.

12. SensorToImage. *SVDK Hardware User Guide, revision 1.1.* SensorToImage, 2015.
13. Priyanka Sharma and Manavjeet Kaur. Classification in pattern recognition: A review. *International Journal of Advanced Research in Computer Science and Software Engineering*, 3(4):3, 2013.
14. Yuri Marchetti Tavares, Nadia Nedjah, and Luiza de Macedo Mourelle. Co-design system for template matching using dedicated co-processor and particle swarm optimization. In *Circuits & Systems (LASCAS), 2017 IEEE 8th Latin American Symposium on*, pages 1–4, Bariloche, Argentina, 2017. IEEE.

A parallel BSO Metaheuristic for Molecular Docking problem

H. Saadi[1], M. Mehdi[2], N. Nouali Taboudjemat[1], H.Benboudjelthia[2], and S.Ousmer[2]

[1] CERIST Research Center Algiers, Algeria
saadi@cerist.dz
n.nouali@cerist.dz

[2] Universite des Sciences et de la Technologie Houari Boumediene, Algiers, Algeria
malika.mehdi@gmail.com
ousmer.sabrine@gmail.com
hafidabenbou@gmail.com

Abstract. In this paper we propose a parallel model of bees swarm optimization metaheuristic to solve the molecular docking problem. This solution is based on the MapReduce model, we use the MapCG framework to implement this model on graphics processing card GPUs. MapCG was developed to simplify the programming process on GPU and design portable application independently of the hardware architecture. Our solution can run sequentially on CPU, or in parallel on GPU without changing the code. Experiments when docking a set of protein-ligand complexes show that our solution achieves a good performance. Indeed, the parallel implementation using MapCG on GPU gains an average speedup of 10x with respect to a single CPU core.

1 Introduction

Molecular docking methods play an important role in the field of computational chemistry and biomedical engineering [1], they are frequently used to predict the binding orientation of small molecule to their protein target in order to predict the affinity and activity of the small molecule measured by a scoring function [2] . Molecular docking gives detailed description of protein-ligand poses in chemical details, provides information on possible binding site, bound conformation and binding strength, which is referred to as binding mode [3]. In protein-ligand docking, an optimization algorithm (Research algorithm) is used to find the best binding pose of the ligand against a target protein by traveling through the search space. This algorithm plays a central role in determining the docking accuracy [4]. The scoring function models chemical interactions and determines the free energy for given poses [5], it gives information about the stability of protein-ligand complex.

Through profiling, in docking simulations most of the CPU time is spent in the evaluation phase (Scoring function) of the metaheuristic [6]. In this phase, the calculation of the soring function consumes more than 80 % of the total execution time and considered as a big challenge in the performance of the metaheuristic. A good solution for this bottleneck is the parallelization of this calculations on the GPUs in order to speedup the docking process and face this challenge [7].

Graphics Processing Units (GPU) have been playing an important role in the general purpose computing field recently. The common approach to program GPU today is to write GPU specific code with low level GPU APIs such as CUDA [8]. Although this approach can achieve very good performance, it raises serious portability issues: developer are required to write a specific version of code for each potential target architecture. It results in high development and maintenance cost.

A good solution for this issue is to use frameworks which facilitate the programing task and provide a portable code which is independent of hardware features.

In this paper we propose a parallelization of *BSO* based metaheuristic for molecular docking problem with MapCG framework [9], this framework in its turn is based on MapReduce [10] programing model which facilitates the task for parallel programming, the aim is to propose portable application which can run on CPUs or GPUs, or in both CPU and GPUs.

The remainder of the paper is structured as follows. In section 2, we present the background needed to understand the remaining sections. To that purpose, we first briefly introduce the MapReduce model which we have chosen as the parallel model, then we introduce the MapCG framework used to implement this model, and finally we summarize some elements of BSO metaheuristic.

In section 3 we introduces our GPU parallelization strategies and in section 4 we give details on experiments and discuss the obtained results. Finally, in section 5 we summarize the results and present some concluding remarks and work perspectives.

2 Background

2.1 MapReduce

MapReduce is a parallel programming model introduced by Google for large data processing on distributed systems. Google used this framework for the purpose of serving Googles Web page indexing, it substitutes earlier Google's indexing algorithms. Developers find the MapReduce model easy to use and beneficial to create parallel programs without any worries about inter-machines communication. It facilitates the task of the the programmer. The user defines the computation in terms of a Map and a Reduce functions, and the associate runtime system accordingly parallelizes the computation across different nodes of cluster or across different CPUs and GPUs cores of the same machine, this runtime system deals with machine collapse by fault tolerance mechanism due to its distributed system nature, it also takes care of the inter-machine communication to better manage the resources. [11].

The Map and the Reduce functions are written by the user, The Map allows to divide and distribute the work on different nodes of the cluster and produces a set of intermediate key/value pairs for each data read. The associated library puts together all intermediate values which have the same intermediate key K and passes them to the reduce function. The reduce function gets the intermediate key provided by the Map function and all values for that key K. It incorporates these values together to form results with possibly one output[10].

In general, here are the steps to follow to write a MapeReduce program:

1- Choose an approach to split the data so that it can be parallelized by the Map.

2- Choose a good key to use for the problem.

3- Write the program for Map function.

4- Write the program for the Reduce function.

Fig. 1 explains an example of text treated in parallel with the MapReduce Model, the aim is to find the number of occurrences of words in a this text file. First, the lines of the file are split into blocks. Then, in the "Map" phase, keys are created with an associated value, in this example a key is a word and the value is the number 1 to indicate that the word is present once. Then, all the identical keys are grouped together (same words).Finally, and in the Reduce phase, a treatment is performed on the all values of the same key (in this example, the values are added together to obtain the number of occurrences of each word).

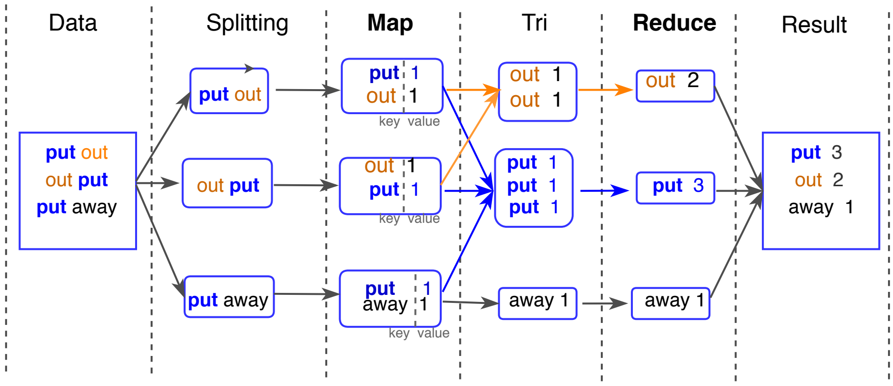


Fig. 1. An example of a text file treated with MapReduce Model

2.2 MapCG

MapCG is a framework based on MapReduce model, It allows programmers to write a parallel program that can be executed on CPU and GPU, the programmer only need to write one version of program, the program contains esentialy the Map and Reduce functions. The framework generates automatically the CPU and GPU versions of the Map and Reduce functions by source code translation, and then it uses the MapCG runtime library to execute them on CPU or/and GPU, this operation ensure portability with a high level of abstraction [8].

Figure 2 shows the MapCG framework architecture. the later contains two parts. The first part, provides the programmers a unified, high level parallel programming environment which allows him to write a MapReduce code once. While the second part represents the MapCG runtime which executes MapReduce code efficiently on different platforms, and bridges the gaps of different hardware features.

In the execution step the input data is split by the *Splitter()* function into pieces, then these pieces are passed to the Map function. The Map function process the data and emits intermediate pairs (key/value) using MapCG *emit intermediate()* function. The intermediate pairs are then grouped and passed to the Reduce function, which emits data using the MapCG *emit()* function. The data emitted by Reduce can then be obtained by invoking the MapCG *get output()* function.

A hash table is used to group the key/value pair on GPU, it is a difficult to implement this table on GPU, because the data must be dynamically allocated. for this purpose MapCG use its one memory allocation system to dynamically allocate memory on GPU and use closed addressing hash table. An other problem is the concurrent insertion issue, MapCG framework uses lock-free algorithm to solve this problem, This algorithm guarantees that the insertion never gets blocked by any particular thread [12].

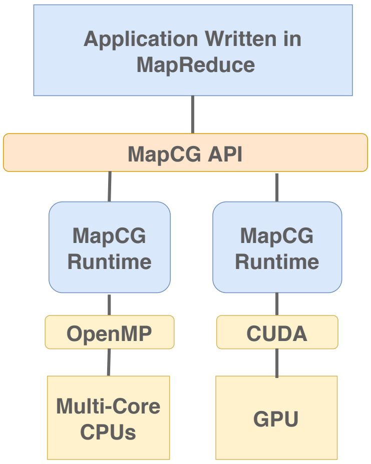


Fig. 2. MapCG architecture overview

2.3 Introduction to BSO Bees Swarm Optimization

The metaheuristic BSO proposed in [13] is inspired by the collective bees behavior. It is based on a swarm of artificial bees cooperating together to solve a problem. First, a bee named InitBee settles to find a solution presenting good features. From this first solution called Sref we determine a set of other solutions of the search space by using a certain strategy. This set of solutions is called Search Area. Then, every bee will consider a solution from the Search Area as its starting point in the search. After accomplishing its search, every bee communicates the best visited solution to all its neighbors through a table named Dance table. One of the solutions stored in this table will become the new reference solution during the next iteration. In order to avoid cycles, the reference solution is stored every time in a taboo list. The reference solution is chosen according to the quality criterion. However, if after a period of time the swarm observes that the solution is not improved, it introduces a criterion of diversification preventing it from being trapped in a local optimum. The diversification criterion consists to select among the solutions stored in taboo list, the most distant one. The algorithm stops when the optimal solution is found or the maximum number of iterations is reached.

3 Methodology

In this section, we present the design and implementation of the parallel strategy of BSO metaheuristic. We first describe the overall design with MapReduce model, and then present the implementation with MapCG framwork on the GPU.

All the BSO search algorithm steps such as the determination of the reference solution, the determination of the regions, the neighbor search, all those steps are executed sequentially on the CPU because their complexity is negligible compared to the evaluation step (the scoring function). the later is parallelized using the MapReduce model in which we calculate the free energy on the GPU efficiently by evaluating several solutions in parallel with MAP workers. The main idea is to Maper the dance table which contains solutions into several pieces, then chose the best solution within the all solutions found in the first step by the reduce function.

Fig.3 explains our parallel approach details, we proposed to parallelize the BSO algorithm, exactly the evaluation step, initially the dance table generated by the neighborhood step is split into several pieces Mi (Maper i) using the split function provided by the MapCG framework, the table dance contains all the solutions generated for on iteration. The number of Mi is determined by this function split based on GPU capacity (number of cores) and data size. Each piece Mi is passed to a map function (a map worker) to evaluate the solutions of each piece. During processing, each map function produces intermediate pairs (key, value). The key represents the number of Mi, and the value represents a solution S which contains the position, the orientation and the conformation. We added two elements to the vector of each solution: an index of the solution, and the energy associated for this solution calculated during this step. A solution S is represented as follows:

S = (Index, Translation, Orientation, Rotation, Energy).

These pairs (key, value) are sent by the *intermediate_function* to the reduce function. In this reduction step the *MIN* function is applied to define the best solution with the smallest energy value of each piece Mi using the reduce function. Finally, the pairs generated during the reduction step are emitted by the *emit* function which calls the *get output* function in order to obtain the solution that has the best energy value for all Mi pieces by choosing the min of the of the different pairs energy.

The pseudo code of the map and reduce functions is given in Algorithm 1:

Algorithm 1 The Map and Reduce pseudocode for the evaluation step BSO

```
Map (String: Key, String: Value) :
count i [K]
for i = 0 to M i do
    // Mi is a subset of solutions
    Evaluate (Si) solutions
end for
emit_intermediate(...)
Reduce (String: key, Iterator * value)
count i [K]
for j = 0 to K do
    Choose the solution with minimum energy()
end for
emit(count i)
```

4 Experiments and results

This section shows the experimental results. We compare the performance of the BSO algorithm implemented in sequential with the parallel version of implemented with MapCG framework.

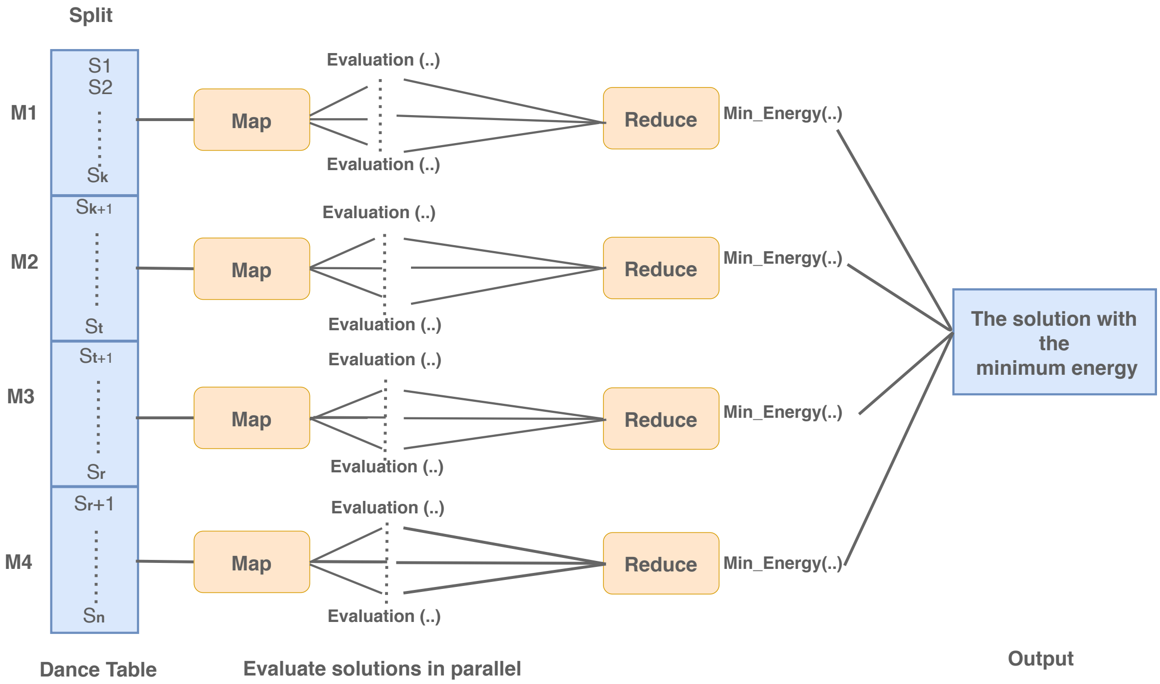


Fig. 3. Parallel BSO design with MapReduce Model

First, we describe the dataset and the environment used for the evaluation before we show the experimental results. We use GeForce GT 740M card, this GPU is a 1.03 GHz processor with 384 cores and 2 GB of memory size. For the CPU side we use Intel Core i5 model, which is 4 GHz processor with 4 cores. On the software side, We use C++ with MapCG framework, we run this experiments on Linux Mint 17.1 64 bits which was chosen for its stability and performance.

The results of this experimentation depend on two factors, the size of the data (large, medium, or small) and the number of iterations of the BSO Metaheuristic. For the size of data we have two elements that increase the complexity and therefore affect the results of our experimentation: First the number of flexible residues on ligand which is known as degree of freedom. Then, the number of atoms in the protein and the ligand, to calculate the energy of interaction between the ligand and the protein we must calculate the energy between each atom of a ligand with each atom of the protein for n iterations.

The number of solution generated depends on the size of the vector which represents the solution. the number of variable for the Position and Orientation fields is constant. However, it is variable for the rotation bounds field (flexible residue). So the number of solutions depends on the number of the flexible bounds in some residues, this later will vary the number of regions (bees) and the population generated by BSO search algorithm. Table 1 describes the population size according to the number of flexible residues bound, here the number of iterations is fixed at 3 iteration.

For the benchmark we use a set protein called *Cutinase* and a ligand called *3QPApropre* from RCSB protein data bank (lien) "http://www.rcsb.org/". The protein has 63 atoms while the ligand has 40 atom and 4 flexible bounds. In this experiment we vary the number of iteration of the BSO algorithm and calculate the execution time for the sequential and parallel version of BSO, then we calculate the speedup obtained with the parallel version on GPU with respect to the sequential execution on CPU.

Table 2 show that the Parallel BSO algorithm outperforms the sequential algorithm, especially for large data sizes where we evaluate a huge number of solutions. We get better occupancy of the GPU when we increase the number of iteration. We obtained a speedup of 10 X with respect to the sequential algorithm.

In our tests we vary the number of flexible ligand bound, and we fix the number of iterations of the BSO algorithm to 3, we get the following execution times in Table 2. Indeed, the performance

Table 1. Population size according to data size.

Degree of freedom	regions size	dance table size
2	192	36288
4	384	145452
5	480	226800
6	576	326592

of the parallel algorithm with MapCG framework on GPU is better than the sequential algorithm on CPU, especially for a large data size, in which evaluate several solutions.

Table 2. performance comparison between sequential and parallel BSO and Speed Up.

Data Size	sequential	parallel	Speed Up
small	140.91	15.51	9.08
medium	550.34	60.21	9.15
large	1236.23	115.51	10.7

5 Conclusion

In this article we proposed a new parallel approach for BSO based metaheuristic to solve the molecular docking problem. This approach is based on Map/Reduce Model. We parallelized and implemented the calculation of the fitness function of BSO on GPUs architecture with MapCG Framework. this framework allow us to write one version of code which can be executed on both CPU and GPU without changing any line of code. The results show a total speed-up exceeding 10x for the evaluation stage with respect to one CPU version, The MapsCG framework and the Map Reduce model are a potentially fruitful area for future research in meta-heuristic parallelizing on GPUs and CPUs, and a good tool to accelerate the docking process which can help in drug design speedup process.

References

1. S. F. Sousa, P. A. Fernandes, M. J. Ramos, Protein–ligand docking: current status and future challenges, Proteins: Structure, Function, and Bioinformatics 65 (1) (2006) 15–26.
2. B. Mukesh, K. Rakesh, Molecular docking: a review, Int J Res Ayurveda Pharm 2 (2011) 746–1751.
3. V. Namasivayam, R. Günther, Pso@ autodock: A fast flexible molecular docking program based on swarm intelligence, Chemical biology & drug design 70 (6) (2007) 475–484.
4. L. Guo, Z. Yan, X. Zheng, L. Hu, Y. Yang, J. Wang, A comparison of various optimization algorithms of protein–ligand docking programs by fitness accuracy, Journal of molecular modeling 20 (7) (2014) 2251.
5. I. Pechan, B. Feher, Molecular docking on fpga and gpu platforms, in: Field Programmable Logic and Applications (FPL), 2011 International Conference on, IEEE, 2011, pp. 474–477.
6. J. Fang, A. L. Varbanescu, B. Imbernon, J. M. Cecilia, H. E. P. Sánchez, Parallel computation of non-bonded interactions in drug discovery: Nvidia gpus vs. intel xeon phi., in: IWBBIO, 2014, pp. 579–588.
7. B. Imbernón, J. Prades, D. Giménez, J. M. Cecilia, F. Silla, Enhancing large-scale docking simulation on heterogeneous systems: An mpi vs rcuda study, Future Generation Computer Systems 79 (2018) 26–37.
8. C. Hong, D. Chen, W. Chen, W. Zheng, H. Lin, Mapcg: writing parallel program portable between cpu and gpu, in: Proceedings of the 19th international conference on Parallel architectures and compilation techniques, ACM, 2010, pp. 217–226.

9. L. Liu, Y. Zhang, M. Liu, C. Wang, J. Wang, A-mapcg: An adaptive mapreduce framework for gpus, in: Networking, Architecture, and Storage (NAS), 2017 International Conference on, IEEE, 2017, pp. 1–8.
10. J. Dean, S. Ghemawat, Mapreduce: a flexible data processing tool, Communications of the ACM 53 (1) (2010) 72–77.
11. J. Dean, S. Ghemawat, Mapreduce: simplified data processing on large clusters, Communications of the ACM 51 (1) (2008) 107–113.
12. C.-T. Hong, D.-H. Chen, Y.-B. Chen, W.-G. Chen, W.-M. Zheng, H.-B. Lin, Providing source code level portability between cpu and gpu with mapcg, Journal of Computer Science and Technology 27 (1) (2012) 42–56.
13. H. Drias, S. Sadeg, S. Yahi, Cooperative bees swarm for solving the maximum weighted satisfiability problem, in: International Work-Conference on Artificial Neural Networks, Springer, 2005, pp. 318–325.

One-Class Subject Authentication using Feature Extraction by Grammatical Evolution on Accelerometer Data

Stefano Mauceri, James Sweeney, James Mcdermott
University College Dublin UCD, Ireland

Abstract. In this study Grammatical Evolution (GE) is used to extract features from accelerometer time-series in order to increase the performance of a Kernel Density Estimation (KDE) classifier. Time-series are collected through nine wrist-worn accelerometers assigned to as many subjects. The goal is to distinguish each subject from all the others in a one-class classification framework. GE-evolved solutions, referred to as feature extractors, are thoroughly analysed. Each solution is a function able to target a specific subsequence of a time-series and reduce it to a single scalar. The choice of subsequence and method of reduction are represented by a flexible grammar. In this way a large time-series can be summarised to an arbitrary number of features. Results show that the proposed evolutionary algorithm outperforms two strong baselines.

1 Introduction

In the field of subject authentication the aim is to confirm the identity of an individual. There are several authentication technologies and applications [22] however in this study we are interested in accelerometer data based subject authentication. In other words, we wish to develop a method which confirms that the accelerometer data produced by a source device corresponds to the individual to which the device is assigned.

In clinical research, scientists would like to use accelerometer data for monitoring the efficacy of treatment options on movement disorders or the impact of drugs on subjects' free living activity levels [14]. However, first they are in need of classification models to confirm that a given device is worn only by the intended subject for the whole trial period. Both errors and misconduct could invalidate studies of considerable cost and duration. In addition, accelerometer data are used to perform activity recognition [20] but this application is not investigated in this study.

Accelerometer data come as a sequence of time-ordered real values, thus the problem we treat falls within the time-series classification domain: given a set of accelerometer time-series we want to be able to separate those that are generated by a specific subject from those that aren't. We address this problem as a feature-based time-series one-class classification problem.

In one-class classification we aim to learn a concept by only using examples of the same concept [13]. As a matter of fact, to distinguish an apple from another type of fruit humans do not need to be trained on all the types of fruit of the planet. It is sufficient to see a few examples of the "class" apple to learn what is an apple and separate it from what is not. In this study we want to separate the accelerometer time-series generated by a subject from those that aren't; we infer a suitable decision rule from a set of accelerometer time-series all coming from that one subject.

In related work (citation redacted) we propose a feature-based time-series classification method which shows good performance. A set of 25 features is manually selected from the statistical and the time-series domains. An accelerometer time-series recorded over an entire day is divided into 24 equally sized subsequences (one per hour). The 25 features are extracted from each subsequence and collected in a feature vector (of length 25×24) then used for classification. This approach, although effective, shows some limitations. For instance, how can we know in advance in how many subsequences it is better to divide a time-series? How can we know if we need to focus on a single subsequence that spans from a to b or on the entire time-series? Again how can we know if for a given subsequence $[a : b]$ it is important to know its mean value or the result of a less intuitive function like *log(mean(sin(x)))*?

These considerations justify the need for a flexible feature extraction method for time-series classification. In terms of flexibility Grammatical Evolution (GE) [18] stands out as a powerful learning framework. In fact, GE allows us to set the general rules for the creation of suitable solutions in a bespoke grammar. Therefore, a grammar guided intelligent search leads to the development of open-ended solutions which can allow the discovery of subtle or complex relationships in the mapping from data to class labels that would be difficult for humans to identify.

The purpose of this work is to evaluate strengths and weaknesses of an automated feature-based time-series one-class classification algorithm implemented through GE. In practice, we use GE to create one or more feature extractors each of which can select a subsequence of a time-series and reduce it to a single scalar (i.e. a feature). A subsequence is reduced to a scalar thanks to a symbolic model composed of functions which naturally "synthesise" a series of numbers into a single one e.g. the mean. In this way a time-series can be reduced to an arbitrary number of features. Such a feature-based representation is meant to maximise the performance of a Kernel Density Estimation (KDE) classifier in the solution of the aforementioned subject authentication problem.

All the features are evolved sequentially. The evolution of the first feature is driven by the search for better classification performance, in terms of AUROC. Subsequent features are evolved according to two objectives: the classification performance to be maximised, and the average coefficient of determination, R^2, with previous features to be minimised in order to reduce linear dependencies between features. This algorithm shows an excellent classification performance in terms of AUROC using only two features.

The layout of the paper is as follows: related work is presented in Section 2. Data is described in Section 3. Section 4 describes the evolutionary system, and the grammar in use. The experiment design is illustrated in Section 5. Results are discussed in Section 6. Finally, some conclusions are drawn in Section 7.

2 Related Work

A time-series is a collection of equally spaced measurements over time. Such data are extremely common across a variety of scientific fields [6]. There are three major approaches to time-series classification: 1) feature-based classification, in which a time-series is summarised in a feature vector; 2) distance-based classification, where one or more distance functions are used to measure the distance between two time-series; 3) model-based classification, where the focus is on the underlying model generating the time-series [24].

Feature-based methods are less sensitive to the amount of noise in the data, facilitate working with time-series of large size, do not require all time-series to have the same length as long as the same number of features is extracted, and allow highlighting of data local and global properties [17, 23]. However, feature extraction for time-series classification is a domain-specific and time consuming process which requires a long trial and error procedure, especially when a new problem is addressed [5]. Nonetheless, the choice of features is central to the feature-based time-series classification problem since it can make the difference in terms of classification performance [10].

While some of the information contained in a time-series can be redundant/irrelevant and negatively affect the predictive performance of learning models [1], how to divide a time-series in a set of highly descriptive subsequences is not known a priori [2]. The use of equally sized subsequences is an oversimplified approach to this problem hence a dynamic method is preferable [4].

GE [18] is a grammar-based approach to Genetic Programming (GP) [15] which evolves human-readable code according to rules specified by a grammar. It can be useful when the modeler has a weak understanding of the relation between the explanatory and the dependent variable [3].

GP is used for feature selection in EEG signals classification [11], speaker verification [16] and selection of the best tuple feature extractor/classifier for fault detection [12]. GE is used to create a non-linear mapping of pre-existing features for fetal heart rate classification [9]. Although related to feature extraction feature selection is a different problem. GP is also used for feature extraction in image classification [21] which is a different application domain.

In a related article considering feature-based time-series classification [5], instead of evolving a population of solutions a single one made of a set of grammar-generated feature extractors is maintained. Each feature extractor reduces a subsequence of a time-series to a single scalar. This hill-climbing algorithm does not make use of any crossover operator but allows modification to the current solution (inclusion, dismissal, mutation of feature extractors) only if changes increase classification accuracy or cause a negligible change in accuracy but a decrease in runtime [5].

To the best of our knowledge the present study represents a novel population-based GP approach to the feature-based time-series one-class classification problem and the subject authentication problem from accelerometer data.

3 Preliminaries

This section introduces the dataset in use (Section 3.1), the data preparation steps (Section 3.2) and the training, validation and test sets design (Section 3.3).

3.1 Dataset Description

Data are collected through wrist-worn tri-axial accelerometers able to measure linear acceleration within a range of ± 16g per axis[1]. The magnitude of the acceleration along the three axis is calculated as the square root of the sum of the square of the single accelerations and rounded to the closest integer. The time-series object of this study are indeed a sequence of magnitude values at the resolution of one data point per minute over an entire day (24 hours). A graph of magnitude recordings for one typical day is shown in Figure 1. All the variables recorded are shown in Table 1. ID is the subject unique and anonymous identifier. Axis_1,2,3 show acceleration along the respective axis.

A sample of 9 volunteers (not enrolled in a clinical trial), composed of 3 females and 6 males, is required to wear the mentioned device for a period of approximately 40 days in free living conditions. All the participants are office workers based in the same location and working from Monday to Friday. Only weekdays are used in this study because weekend days are relatively few.

ID	Subject_1
Date	2016-03-21
Weekday	0
Hour	11
Minute	36
Axis_1	168
Axis_1	544
Axis_1	563
Magnitude	801

Table 1: Dataset variables.

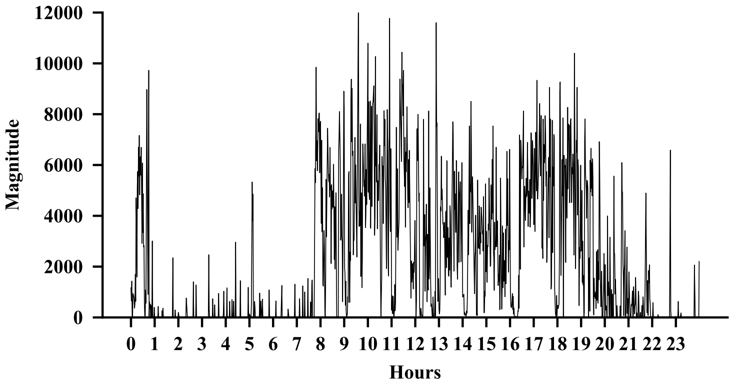


Figure 1: Magnitude recordings for one day for one subject.

3.2 Data Preparation

The algorithmic implementation in use requires a full day of recordings i.e. each time-series has to contain 1440 observations (24 hours × 60 minutes). The original dataset is filtered as follows:

- Time-series with a cumulative magnitude lower than 500,000g are removed. These time-series predominantly contain 'non-wear time' i.e. time when the device is on but it is set down and stationary. In such a scenario a device records only a long sequence of zeros. Note that this does not exclude the possibility to find some 'non-wear time' in the remaining time-series.
- Some time-series may contain a sequence of missing values i.e. they contain less than 1440 observations. This is caused by the fact that from time to time it is necessary to download the data from the limited internal memory of the device (4GB) and recharge its batteries. In order to deal with this, when a sequence of contiguous missing values of size N is found it is filled with a copy of the N values that precede it. The maximum length for a sequence of missing values allowed is 90 minutes; if this limit is exceeded the time-series is discarded. This strategy for handling the missing data may seem naive but it is adopted for its simplicity and because it is required only two to three times per subject.

[1] For further information see: http://www.actigraphcorp.com/.

3.3 Training, Validation and Test Sets

After filtering the dataset as described in Section 3.2 the amount of time-series available is different from subject to subject. The subject with the fewest has 23 of them. From those subjects where a larger number is available, 23 are randomly selected.

For each subject, 18 of the 23 time-series are included in the training set, while the remaining 5 are included in the test set. Thus a training set of 162 time-series (18 time-series × 9 subjects) and a test set of 45 time-series (5 time-series × 9 subjects) are obtained. A validation set is created using a 2:1 split of the training data maintaining a constant number of time-series per subject.

4 Evolutionary System

The overall evolutionary system is described in Section 4.1. The grammar in use is discussed in Section 4.2. The system is implemented in Python and relies on the PonyGE2 library [7]. The KDE classifier uses Scikit-Learn [19]

4.1 System Overview

The evolutionary process involves one subject at a time, thus evolved features are specifically developed to have a good classification performance for that one subject.

All the elements of a population of GE-evolved feature extractors are evaluated one at a time. The evaluation process is depicted in Figure 2. A feature extractor is used to select a subsequence of a time-series and reduce it to a scalar (i.e. a feature). This is done for all the time-series in the training and the validation sets. A KDE classifier is fit on the feature-based representation of the training set and than used to estimate the probability of each instance of the feature-based representation of the validation set to belong to the target class. Estimated probabilities are used to calculate the AUROC.

For the first feature, each feature extractor is assigned a fitness score (i.e. a measure of quality) equal to the classification error: (1−AUROC score). Subsequent features are assigned a fitness equal to the sum of the classification error plus the average coefficient of determination, R^2, of the current feature with previous ones. As an example, if we are evolving the 3$^{\text{rd}}$ feature, we first calculate the R^2 between the feature-based representation of the training set according to the current feature extractor and each of the feature-based representations of the training set according to the feature extractors previously evolved. Than we calculate the average R^2 and sum it to the classification error of the current feature extractor.

The system works to minimise the classification error (i.e. maximise the AUROC score) of the current feature extractor, and to minimise the R^2 with previous features in order to reduce linear dependencies between features. When the evolutionary process is completed the feature extractor with the lowest fitness is considered the final solution of the algorithm and tested on a portion of unseen data. We use a KDE classifier because it is a simple and computationally efficient model, however any other one-class classifier could be plugged into the system instead.

4.2 Grammar

A grammar in the Backus-Naur form, as shown in Figure 3, guides the creation of a feature extractor. The mapping starts from the production rule `<f_ext>`. In `<f_ext>` there are two production choices both intended to select a subsequence from a time-series and reduce it to a single scalar. A subsequence is selected using a lower bound (`<lb>`) and an upper bound (`<ub>`) randomly generated with uniform probability from the range [1 : 1440]. Since a requirement is that the lower bound is below the upper bound, if this condition is violated they are swapped. If the lower bound is equal to the upper bound than the latter is increased by 1.

As shown in Figure 4, once bounds are set a subsequence can be treated according to two different strategies: `Select` and `Select_fold_n_times`. The `Select` function picks up a subsequence and than applies a function `<f>` on it which causes the reduction to a single scalar. The `Select_fold_n_times` function picks up a subsequence and fold it in `<n>` sub-subsequences. A function `<f>` is applied on each sub-subsequence that in this way is reduced to a single scalar.

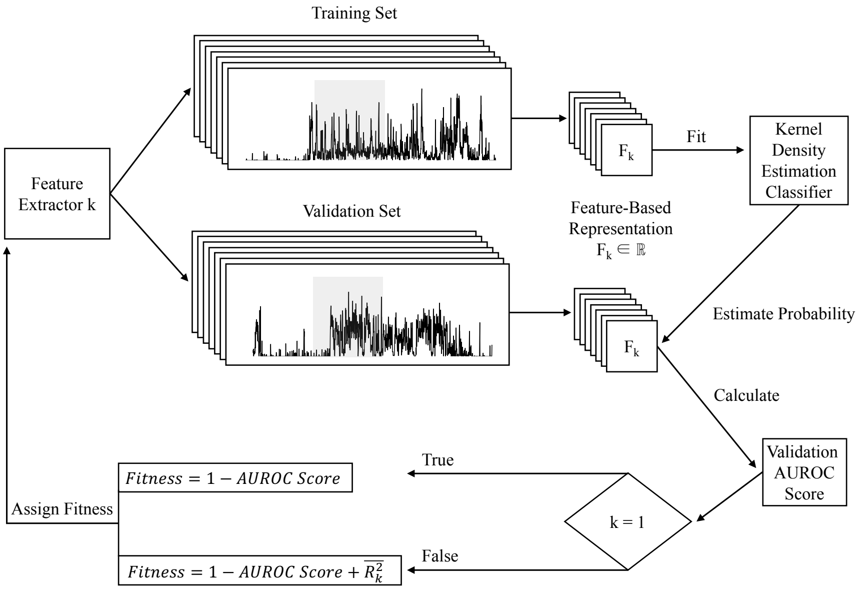

Training Set
Feature Extractor k
Validation Set
F$_k$
Fit
Kernel Density Estimation Classifier
Feature-Based Representation F$_k$ ∈ ℝ
Estimate Probability
F$_k$
Calculate
Validation AUROC Score
True
$Fitness = 1 - AUROC\ Score$
Assign Fitness
k = 1
False
$Fitness = 1 - AUROC\ Score + \overline{R_k^2}$

Figure 2: Evolutionary system summary: evaluation of a feature extractor. $\overline{R_k^2} = \frac{1}{k-1}\sum_{i=1}^{k-1} R^2(F_i, F_k)$.

```
<f_ext>::= Select(<lb>, <ub>, apply=<f>) | Select_fold_n_times(<lb>, <ub>, <n>, apply=<g>)
<p> ::= (<t>+<t>) | ... | Moving_Average(<t>, <n>)
<f>::= Mean(<t>) | ... | Entropy(<t>)
<g>::= Mean(<f>) | ... | Entropy(<f>)
<t>::= X | <p>
<lb>::= random[1, 1440]
<ub>::= random[1, 1440]
<n>::= random[1, 20]
```

Figure 3: Grammar structure. All the functions in use are listed in the text.

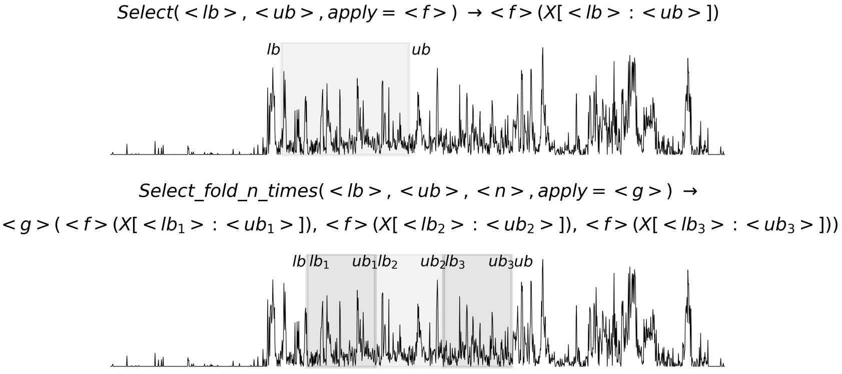

$Select(<lb>, <ub>, apply=<f>) \rightarrow <f>(X[<lb> : <ub>])$
lb
ub
$Select\_fold\_n\_times(<lb>, <ub>, <n>, apply=<g>) \rightarrow$
$<g>(<f>(X[<lb_1> : <ub_1>]), <f>(X[<lb_2> : <ub_2>]), <f>(X[<lb_3> : <ub_3>]))$
lb lb_1 ub_1 lb_2 ub_2 lb_3 ub_3 ub

Figure 4: Subsequence selection strategies.

Resulting values are reduced to a single one thanks to a function `<g>`. If a subsequence can't be folded in `<n>` equally sized sub-subsequences the upper bound is adjusted according to |`<lb>`−`<ub>`| mod `<n>`.

Both the production rules `<f>` and `<g>` include the same functions: mean, standard deviation, variance, median, mode, skewness, kurtosis, max, min, sum, autocorrelation, mean absolute deviation and entropy. All these functions are included because they are able to reduce a selected subsequence to a single value. Along with the functions in `<p>` these are among the simplest and most commonly used functions in feature-based time-series classification [8].

The set of terminals is defined in `<t>`. It is possible to select a time-series X or a function from the production rule `<p>`. All the functions in `<p>` do not alter the shape of the input they are provided with. In this set are included: four operators $\{+, -, *, /\}$, sine and cosine, logarithm, square root, absolute value, remove linear trend, exponential smoothing, moving variance, moving max, moving min, moving average.

5 Experiment Design

For each subject in the dataset we search for the best features that allow his/her authentication. We sequentially evolve 5 feature extractors per subject, and we repeat this for 30 runs. The optimal number of features required isn't known a priori. However it is expected to depend on the quality of the features and on the problem difficulty meaning that a subject can be authenticated more easily than others.

An increasing number of feature extractors from 1 to 5 are tested on a portion of unseen data. The training and the validation set used during the evolutionary process are merged in a new training set. Both the new training set and the test set are reduced to a certain number of features according to the number of feature extractors at hand. Considering that extracted features can have different scales column-wise standardisation is applied by removing the mean and scaling to unit variance. A KDE classifier with Gaussian kernel is fit on the training set and than used to estimate the probability of the instances in test test to belong to the target class.

5.1 Baselines

GE-evolved feature extractors are compared with the results achieved by a high-performing manual feature extraction method we developed in another work (citation redacted). A second baseline is obtained by passing the entire time-series to the KDE classifier. Furthermore, two baselines are used: 1) most frequent: which always predicts the most frequent label in the training set, 2) uniform: which predicts with uniform probability at random among the labels in the training set.

5.2 Run Parameters

Evolution follows a generational approach with a population of 500 individuals for 40 generations. The population is initialised creating random genomes with uniform probability. The genome has length 250 multiplied by the number of features to be extracted. The maximum derivation tree depth is set at 15. The rest of the parameters are as follows: tournament selection of size 5, int-flip mutation with probability 0.01 per-gene, two-point crossover with probability 0.8, and no wraps are allowed. Elitism is used to preserve the best individual of the population.

6 Results and Discussion

Results for the experiment described in Section 5 are reported in Table 2. Table entries represent the average performance of 30 independent runs of the GE system discussed in Section 4 in terms of AUROC for all the subjects included in our dataset and for all the number of features in the range [1 : 5]. The table also includes the following lines:

- Maximum: maximum performance achieved per subject.
- Manual FE: performance of the manual feature extraction baseline.
- Raw Data: performance achieved by passing the raw time-series to the classifier.
- Uniform: classifier choosing class labels with uniform probability.
- Most Frequent: classifier always choosing the most frequent class label.

Results show how the discussed GE system allows higher performance in terms of AUROC than all the baselines, including the manual feature extraction method we previously developed. While Manual FE achieves an average performance of 78% AUROC using 600 features, the GE system achieves an average performance equal to 87% AUROC using only 2 features. The gap between the two approaches is equal to 9%, and it is greater if the maximum performance per subject over the different number of feature is considered. In this case the GE system achieves an average performance of 89% AUROC.

In Figure 5 is shown the average AUROC score per subject and per number of features. Only subjects 2, 6 and 7 are highlighted because for these subjects the classification performance increases as more features are used. The largest improvement of 12% AUROC, between 1 and 5 features, is related to subject 2. This allows us to hypothesize that the present GE system might benefit from a mechanism that keeps evolving more features until the classification performance stops improving.

In Figure 6, using the first two features evolved from a randomly selected run to provide a 2D representation of data, we show the density estimation of the KDE classifier for subject 1 and 7. The GE system tends to concentrate the instances of the target class around the origin for both features even though this is not explicitly required. This aspect shows that the proposed GE system has the potential to work well using a "simple" radial basis function classifier which has the advantage of requiring only a single hyper-parameter i.e. a distance threshold from the origin.

	Subject									
Features	1	2	3	4	5	6	7	8	9	Avg.
1	98	80	93	83	91	74	64	93	93	86
2	97	86	94	83	90	77	68	94	93	87
3	95	89	91	80	91	79	68	94	89	86
4	95	91	89	78	92	81	71	94	84	86
5	94	92	89	79	92	81	72	94	86	87
Avg.	96	88	91	81	91	79	69	94	89	86
Maximum	98	92	94	83	92	81	72	94	93	89
	Baselines									
Manual FE	94	89	69	67	86	83	65	58	90	78
Raw Data	78	38	78	60	76	64	62	73	66	66
Uniform	52	56	60	81	50	48	61	39	70	58
Most Frequent	50	50	50	50	50	50	50	50	50	50

Table 2: Results and baselines - AUROC scores.

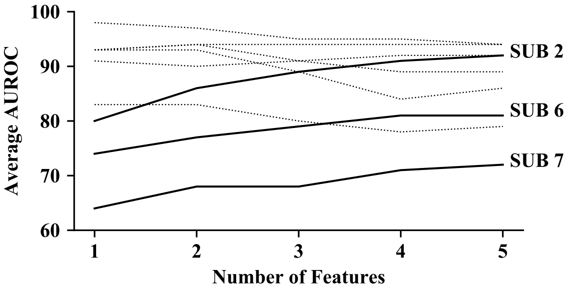


Figure 5: Average AUROC score per subject and per number of features.

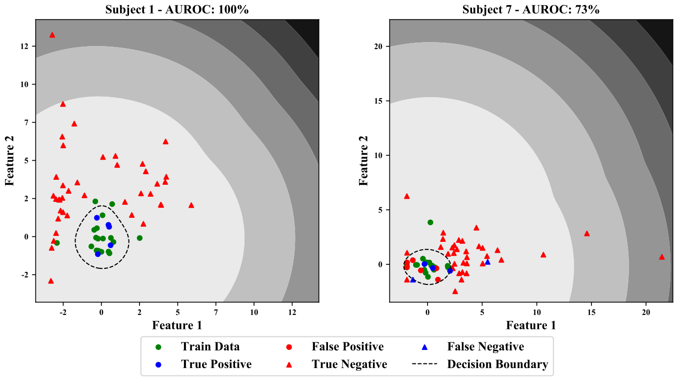


Figure 6: Density estimation of the KDE classifier for subject 1 and 7 using the first 2 features evolved. The decision boundaries shown correspond to the best thresholds for these subjects.

6.1 Frequently Selected Subsequences and Functions

It is interesting to understand whether or not the GE system is preferring the selection of certain subsequences and functions rather than others. Findings are shown in Figure 7 and 8.

In Figure 7, for each subject all the time-series in the training set are averaged and the resulting time-series is presented in black. The red line is obtained by counting how many times a given minute falls within a selected subsequence, over all the features extractors evolved for each subject. Y-axes limits are set according to the maximum either in terms of Magnitude or Selection Frequency found. The first row from the top shows the selection frequencies we would observe if 10,000 lower bounds and 10,000 upper bounds would always be drawn from a uniform distribution. Rows from Sub_1 to Sub_9 show how evolution is driving the selection towards certain subsequences rather than others for each subject. For instance, the hours between 6am and 9am are the most often selected for subjects 1, 2, 4, 5, and 7 but not for the others. Evolution is not only over-selecting certain subsequences rather others, but it is also completely ignoring large sections of the time-series e.g. subjects 1, 2 and 7, in order to focus on restricted areas that enable better classification performance.

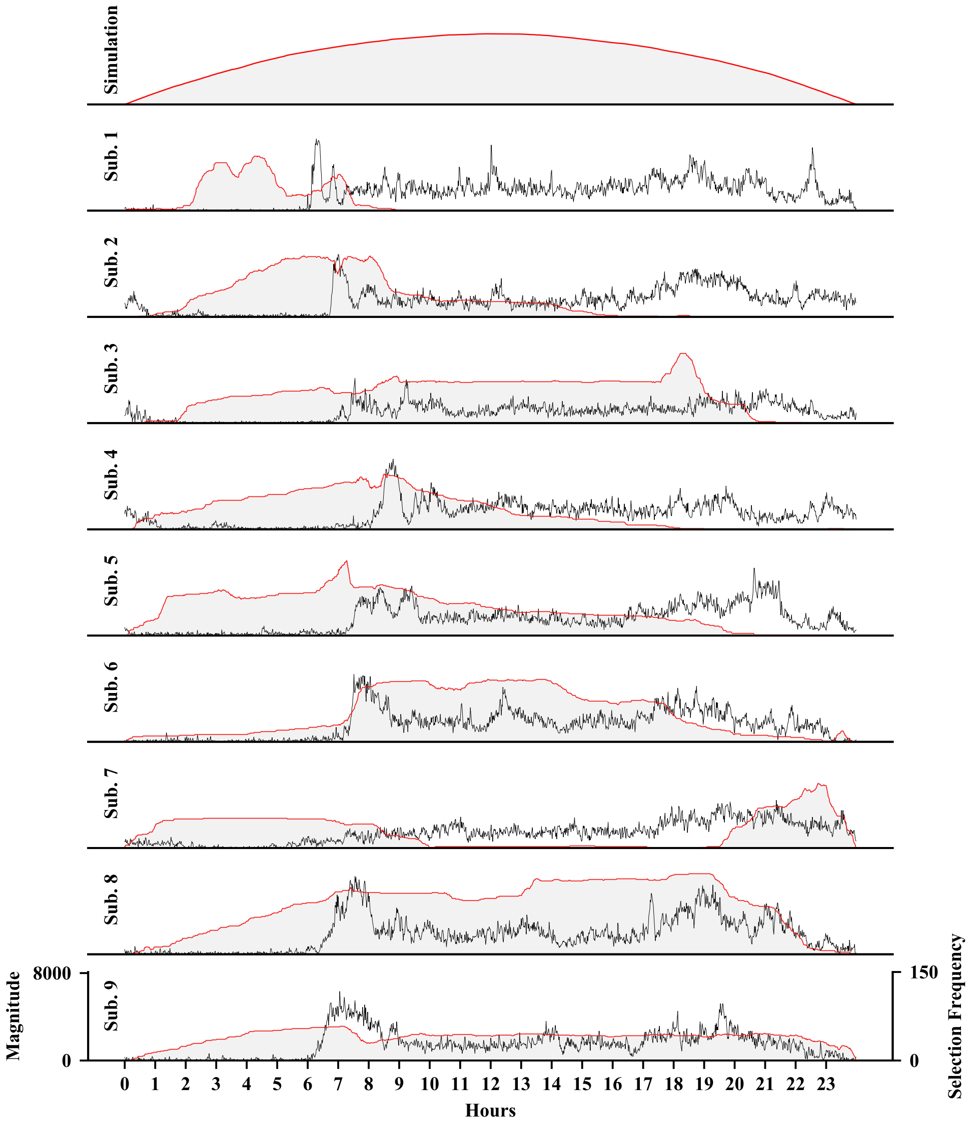


Figure 7: Most frequently selected subsequences per subject.

In Figure 8 we count how many times each function of the grammar is used in a GE-evolved feature over all the features evolved for all the subjects. Counts are expressed as percentage of the total. The average frequency selection is 3.7%. Results show that most frequently selected functions are those that measures the central tendency of data and their variability: mean, standard deviation, variance, and mean absolute deviation. In this set the sum function is included. Preprocessing strategies seem to be important for classification performance; smoothing functions like exponential

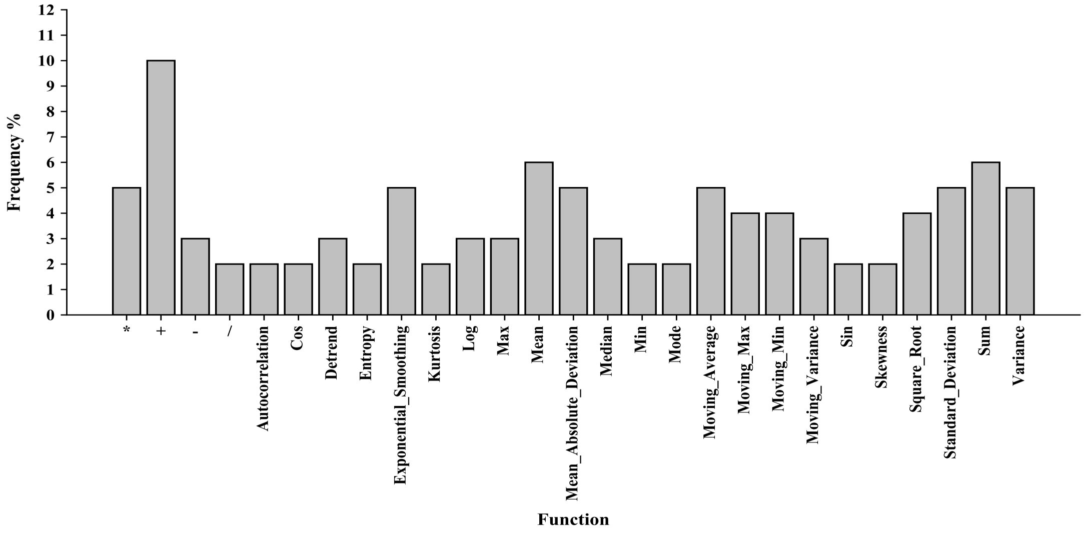


Figure 8: Most frequently selected functions over all subjects.

smoothing, moving average, moving max, moving min are frequently selected. Again, from the set of preprocessing strategies three operators +, $*$ and the square root are selected more frequently than the average. Of these, the + operator is the most selected function.

7 Conclusions

In this study a grammar-guided evolutionary system for subject authentication using accelerometer data is presented. The GE system explores a solution space of feature extractors. Each feature extractor is able to select a subsequence of a time-series and reduce it to a single scalar (i.e. a feature). By choosing the number of features to be evolved a large time-series can be reduced to an arbitrary number of features. Such a feature-based representation is used to authenticate nine subjects in nine distinct one-class classification experiments thanks to a KDE classifier.

The proposed approach is intended to overcome some difficulties encountered in manual feature-based time-series classification methods as discussed in Section 1 and Section 2. The core idea is to relieve the modeler from making any assumptions on how to select subsequences of a time-series and what features to extract from them. It is found that the GE system intelligently drives the search of subsequences and features that enable high classification performance.

A peak in classification performance in terms of AUROC is found using just two features. Results are compared with a manual feature extraction method we previously developed. The current method not only uses two features as opposed to the 600 required by the previous one but also it outperforms its classification performance by 9% (87% AUROC vs. 78% AUROC).

Acknowledgment

Acknowledgments

References

1. D. Bacciu. Unsupervised feature selection for sensor time-series in pervasive computing applications. *Neural Computing and Applications*, 27(5):1077–1091, 2016.
2. E. Bingham, A. Gionis, N. Haiminen, H. Hiisilä, H. Mannila, and E. Terzi. Segmentation and dimensionality reduction. In *Proceedings of the 2006 SIAM International Conference on Data Mining*, pages 372–383. SIAM, 2006.

3. A. Brabazon, K. Meagher, E. Carty, M. O'Neill, and P. Keenan. Grammar-mediated time-series prediction. *Journal of Intelligent Systems*, 14(2-3):123–142, 2005.
4. F.-L. Chung, T.-C. Fu, V. Ng, and R. W. Luk. An evolutionary approach to pattern-based time series segmentation. *IEEE Transactions on Evolutionary Computation*, 8(5):471–489, 2004.
5. D. Eads, K. Glocer, S. Perkins, and J. Theiler. Grammar-guided feature extraction for time series classification. In *Proceedings of the 9th Annual Conference on Neural Information Processing Systems (NIPS'05)*, 2005.
6. P. Esling and C. Agon. Time-series data mining. *ACM Computing Surveys (CSUR)*, 45(1):12, 2012.
7. M. Fenton, J. McDermott, D. Fagan, S. Forstenlechner, E. Hemberg, and M. O'Neill. Ponyge2: Grammatical evolution in python. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pages 1194–1201. ACM, 2017.
8. B. D. Fulcher and N. S. Jones. Highly comparative feature-based time-series classification. *IEEE Transactions on Knowledge and Data Engineering*, 26(12):3026–3037, 2014.
9. G. Georgoulas, D. Gavrilis, I. G. Tsoulos, C. Stylios, J. Bernardes, and P. P. Groumpos. Novel approach for fetal heart rate classification introducing grammatical evolution. *Biomedical Signal Processing and Control*, 2(2):69–79, 2007.
10. H. Guo, L. B. Jack, and A. K. Nandi. Feature generation using genetic programming with application to fault classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 35(1):89–99, 2005.
11. L. Guo, D. Rivero, J. Dorado, C. R. Munteanu, and A. Pazos. Automatic feature extraction using genetic programming: an application to epileptic eeg classification. *Expert Systems with Applications*, 38(8):10425–10436, 2011.
12. D. Y. Harvey, K. Worden, and M. D. Todd. Robust evaluation of time series classification algorithms for structural health monitoring. In *SPIE Smart Structures and Materials+ Nondestructive Evaluation and Health Monitoring*, pages 90640K–90640K. International Society for Optics and Photonics, 2014.
13. H. He and Y. Ma. *Imbalanced learning: foundations, algorithms, and applications.* John Wiley & Sons, 2013.
14. L. A. Kelly, D. G. McMillan, A. Anderson, M. Fippinger, G. Fillerup, and J. Rider. Validity of actigraphs uniaxial and triaxial accelerometers for assessment of physical activity in adults in laboratory conditions. *BMC medical physics*, 13(1):5, 2013.
15. J. R. Koza. *Genetic programming: on the programming of computers by means of natural selection*, volume 1. MIT press, 1992.
16. R. Loughran, A. Agapitos, A. Kattan, A. Brabazon, and M. O'Neill. Feature selection for speaker verification using genetic programming. *Evolutionary Intelligence*, pages 1–21, 2017.
17. A. Nanopoulos, R. Alcock, and Y. Manolopoulos. Feature-based classification of time-series data. *International Journal of Computer Research*, 10(3):49–61, 2001.
18. M. O'Neil and C. Ryan. Grammatical evolution. In *Grammatical Evolution*, pages 33–47. Springer, 2003.
19. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
20. N. Ravi, N. Dandekar, P. Mysore, and M. L. Littman. Activity recognition from accelerometer data. In *Aaai*, volume 5, pages 1541–1546, 2005.
21. L. Shao, L. Liu, and X. Li. Feature learning for image classification via multiobjective genetic programming. *IEEE Transactions on Neural Networks and Learning Systems*, 25(7):1359–1371, 2014.
22. J. Wayman, A. Jain, D. Maltoni, and D. Maio. An introduction to biometric authentication systems. *Biometric Systems*, pages 1–20, 2005.
23. S. J. Wilson. Data representation for time series data mining: time domain approaches. *Wiley Interdisciplinary Reviews: Computational Statistics*, 9(1), 2017.
24. Z. Xing, J. Pei, and E. Keogh. A brief survey on sequence classification. *ACM Sigkdd Explorations Newsletter*, 12(1):40–48, 2010.

A heuristic approach for standalone clinical laboratory layout design

S. Faramarzi Oghani[1,2], E-G. Talbi[1,2], M. Bué[1], E. Varlet[3]

[1] *INRIA Lille-Nord Europe - 40 Avenue Halley, 59650 Villeneuve d'Ascq, France*

[2] *Lille University of Science and Technology - Cité Scientifique,59650 Villeneuve d'Ascq, France*

[3] *Beckman Coulter, Normand-Info - 7 Rue Fréderic Degeorge, CS 10451 - 62028 Arras Cedex, France*

sohrab.faramarzi-oghani@inria.fr, el-ghazali.talbi@inria.fr, martin.bue@inria.fr, evarlet@beckman.com

Abstract: In this paper, a heuristic approach is developed to minimize the total traveling distance within a single-floor standalone clinical laboratory respecting physical, technical and organizational constraints for facility layout problem. This approach proposes various efficient solutions to provide decision makers number of alternatives to opt from. To evaluate the applicability and efficiency of the proposed approach, facility layout of a real standalone clinical laboratory is specified through the algorithm. The output presents feasible and satisfactory results.

Keywords: facility layout problem, heuristic algorithm, clinical laboratory design, healthcare

1 Introduction

Standalone clinical laboratories are mixed systems comprising machines and operators which aim to analyze patients' samples in the form of tubes. A lot of movements take place in such an environment by the operators to move the racks of tubes among different points and to check the test results on validation consoles. Efficient physical placement of instruments in a standalone clinical laboratory not only reduces walking time and distance of the staff within the organization but also provides a convenient working environment by respecting technical and organizational constraints. The physical arrangement of facilities within a system is one of the significant design problems which has a great effect on the efficiency and productivity of the system. This problem is known as *Facility Layout Problem* (FLP) and has been broadly investigated in the literature [1]. Quadratic Assignment Problem (QAP) presented by Koopmans and Beckman is one of the earliest efforts to specify the best position of each facility within a system through assigning the facilities to the pre-specified locations with equal sizes [2]. Heragu proposed nonlinear mathematical models to formulate the FLP considering facilities with equal or different areas [3]. Many variations of the FLP have been introduced in [4]. Complexity of the layout problem is known as NP-hard indicating that exact algorithms are not efficient for medium and large scale instances [3]. Hence, many construction and improvement heuristic algorithms as well as meta-heuristic algorithms have been addressed ([5], [6], [7] and [8]). Construction algorithms are used to create an initial solution while improvement algorithms are applied to improve an initial solution.

This study presents a great endeavour to provide efficient, feasible and applicable alternatives for layout design problem of standalone clinical laboratories respecting the main needs appreciated within such organization. In this paper, a heuristic approach is proposed to deal with the problem of layout design of a standalone clinical laboratory. The rest of this paper is organized as follows: the problem is described in section (2) introducing the significant characteristics of the layout design of a standalone clinical laboratory. Section (3) presents the proposed heuristic approach. In section (4), a real clinical laboratory is considered as a case and its layout is designed through the proposed approach. Section (5) ends the paper with conclusion and future perspectives.

2 Problem description

Generally, facility layout problem deals with the physical arrangement of facilities within an area. Physical arrangement of facilities comprises the specification of location and orientation of each facility to optimize one or more objectives respecting different types of constraints. Specifically, the layout design problem of a standalone clinical laboratory is defined as the specifying the location and orientation of each instrument within the laboratory area to minimize the total traveling distance satisfying physical, technical and organizational constraints. Hereafter, the layout design problem of standalone clinical laboratory is described with more attention to details.

2.1 Definitions and problem assumptions

In this section, related definitions and problem assumptions are discussed in the form of two classes: laboratory area-related and facility-related definitions and assumptions.

2.1.1 Laboratory area-related definitions and assumptions

- Laboratory area is a single-floor place where facilities are arranged. This area is given and can have any shape either simple or complex polygons. To create a real-shape laboratory area, a discretized rectangular area is first considered. Then, extra places are removed to obtain the real shape.

- Position of occupied areas within the laboratory area boundaries is given. In better words, positions occupied by pillars, pre-located facilities, etc. are known, so that no other facilities can be placed there.

- Entrance is a place where tubes enter to the laboratory. Location of the entrance is given. Entrance can be any place among the laboratory area boundaries or even any place in the middle of the laboratory. Generally, for the case where entrance is located in the middle, pneumatic system is used to deliver the arriving tubes.

Figure 1 presents an example of a discretized laboratory area where facilities can be only placed on white cells indicating free places.

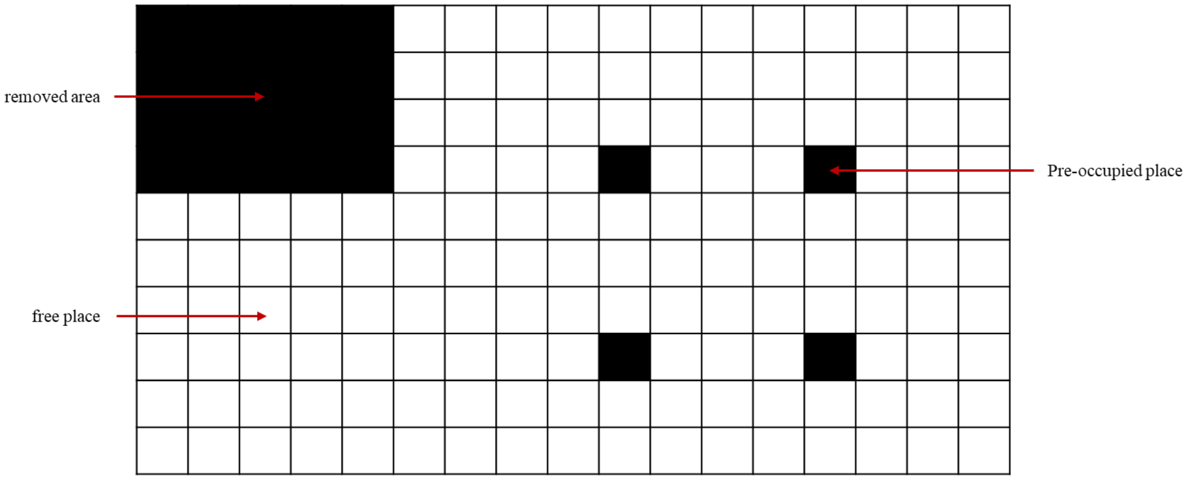


Figure 1. Example for a laboratory area

2.1.2 Facility-related definitions and assumptions

- Facility is a general term referring to workstation, machine, instrument, workbench, etc. In this study, the aim is to efficiently arrange facilities with different sizes within the laboratory area.

- All facilities are considered rectangular with known dimensions. Each facility is discretized based on the discretization scale so that, a facility may occupy one or more free places within discretized laboratory area. Once, a facility is placed within the laboratory area, a cell is considered as the connecting point (I/O) of the facility with the other facilities (called centroid).

- A clearance is defined for each facility. This clearance is simply added to the size of the facility.

- Orientation of each facility is not known as a priori and is determined by th algorithm. It is worth noting that only two orientations are possible for a facility: horizontal or vertical.

- Each facility is associated with a discipline implying that facilities with the same discipline must be placed adjacent enough in one site in a way that at least each facility is neighbour with a facility with similar discipline.

- Distance between centroids of any two facilities is rectilinear distance (Manhattan distance) as it fits more realistic while presenting staff movements between facilities inside the laboratory. Rectilinear distance is computed for two $p(x, y)$ and $p'(x', y')$ points as: $d_{pp'} = |x - x'| + |y - y'|$.

- Relationship between each pair of facilities is demonstrated by a quantitative asymmetric flow matrix. In this study, values of this matrix could present the average number of tubes transported from one point to another or the average number of time that an operator travels between two points.

2.2 Objective function

The objective is to arrange all facilities within the laboratory area in a way that total traveling distance among facilities is minimized. Total traveling distance among facilities is calculated by equation (1):

$$\sum_{e \in E} \sum_{\substack{e' \in E \\ e' \neq e}} f(e, e') * d(c_e, c_{e'}) \qquad (1)$$

Where, E is a set of facilities; e and e' denote the facilities as the members of E; c_e and $c_{e'}$ imply the centroids of facilities e and e', respectively; parameter $f(e, e')$ denotes the flow between the facility pair (e, e'); parameter $d(c_e, c_{e'})$ denotes the distance between the centroids of facility pair $(c_e, c_{e'})$.

2.3 Problem constraints

In this study, problem constraints are discussed in three groups: physical, technical and organizational constraints. Hereafter, each of these constraints is presented:

- *Physical constraints* imply that overlapping is not acceptable between each pair of facilities and between each facility and a pre-occupied place.

- *Technical constraints* deal with the required clearance for each facility which must be respected from technical or safety point of view.

- *Organizational constraints* deal with the adjacency of facilities with similar discipline in the proposed layout; hence, it is also called ‘adjacency constraints’. According to this constraint, all facilities of the same discipline must be placed adjacent enough in one site of the laboratory area in a way that at least each facility in neighbour with a facility with similar discipline. Unlike two other constraints which are mostly common for most of facility layout problems, this constraint is originated from the clinical laboratory organization.

3 Resolution approach

Since facility layout problem lies in NP-hard class in terms of problem complexity, mathematical programming approaches are not efficient for large-scale problems. Hence, in this study, a heuristic approach is developed to tackle facility layout problem in a standalone clinical laboratory. This heuristic has been inspired from Computerized Relationship Layout Planning (CORLAP) algorithm. CORELAP is a qualitative construction algorithm by which an initial solution is proposed for FLP [3]. In this study, an enumerative construction method is firstly applied to reach initial efficient solutions. This method is the integration of a construction method inspired from CORELAB and a type of Branch and Bound (B&B) algorithm. Secondly, initial solutions are compared with the best available solution and consequently, certain number of diverse solutions are excerpted which look differently in the arrangement. Finally, selected solutions are improved through 2-opt algorithm. In brief, the proposed approach is constructed based on the following three steps:

- Generation of initial solutions

- Selection of diverse solutions

- Improvement of the selected solutions

Steps of the proposed heuristic approach is described in the following steps with more attention to details.

3.1 Generation of initial solutions

Basically, generating initial solutions includes two phases: (I) facility sorting, (II) facility placement.

3.1.1 Facility sorting

The first step is to sort the list of all facilities to specify the order of their placement in the laboratory area to build the layout. Basically, this order follows the way material (tubes, information) flows through the system. Here, information flow implies the test results communicated between the analyzers and the corresponding validation consoles and causes operator movement between them. Subsequent rules are used to characterize this order:

1. Facility ‘registration’ lies always in the head of the sorting list.

2. Considering the latest facility added to the sorting list, facility which has the most connections (from and to) with the previous one is then selected. If no connection is found, rule 3 is triggered.

3. Facility with the highest Total Closeness Rating (TCR) is selected. Term ‘TCR’ has been introduced in CORELAP algorithm [3] and it is computed from the flow matrix, as the sum of input and output flows for each facility. In a case of a tie, a facility is selected randomly.

4. Rules 2 and 3 are iterated until all facilities are sorted.

After sorting phase, facilities are selected one after another from sorting list and placed within the laboratory area respecting the existing constraints.

3.1.2 Facility placement

This phase is composed of a construction algorithm combined with a B&B method. The aim of this combination is to generate diverse initial solutions to provide decision makers different alternatives. Following rules must be respected in order to achieve feasible efficient solutions:

1. Facility ‘registration’ which is always first in the sorting list is placed next to the entrance. In fact, ‘registration’ is a place where arriving tubes are registered to the Laboratory Information System (LIS).

2. Once facilities are placed, a neighbourhood is created around the located facilities. This neighbourhood area indicates the possible places to put the centroid of the new coming facility into the layout. Regarding the type of new coming facility, the existing neighbourhood may be reduced to the free places which are next to the pre-located facilities with the same discipline to meet the adjacency constraints. For each place of the neighbourhood, a score is computed considering distance and flow between the new coming facility and the other already placed ones. Note that, the lowest score shows the best place to locate the centroid of the facility.

3. The neighbourhood of the selected place is searched to verify if there is enough space to room the facility or not. If there is sufficient room, then, the facility is placed and the required places are occupied. Finally, the score of the partial solution is updated. Note that this score is the value of the objective function.

4. Rules 2 and 3 are iterated until all facilities are placed within the laboratory area.

In order to generate more initial solutions, the aforementioned placement algorithm is integrated with a B&B method. To do so, instead of placing a facility’s centroid in the best place of the neighbourhood in rule 2, set of suitable places among the neighbourhood is selected, far enough from one another to maintain diversity and to control combinatorial expansion of the tree of possibilities and then, from all of these potential places, a partial solution is created. Different parameters control the expansion of partial solutions enumeration. The first one is the number of possible places in the neighbourhood from which the solution can be extended. The second one is the minimum distance between two selected possible points to do the extension. Furthermore, once a place is selected to locate the centroid of the facility, the facility may take horizontal or vertical orientation. Score of each generated partial solution is computed through equation (1). All the generated partial solutions are extended from the best to the worst one until either they reach to a final solution or terminate with one of the stop criteria. Partial solution expansion is stopped for a branch (the branch is cut) if the score of this solution is worse than another existing solution with more facilities. In addition, once not enough space is available to fit the new coming facility, the branch is terminated.

3.2 Selection of diverse solutions

Most of the generated initial solutions look similar. To provide decision makers efficient solutions with more diversity, solutions are compared based on a diversity measure. This measure is defined as the sum of distances between gravity centers of each discipline in two different solutions.

To avoid comparing each pair of solutions, the best solution is automatically selected. Then, from the second best to the other solutions, solutions which are diverse enough compared to the ones already selected are excerpted. These comparison and selection actions continue until all solutions are investigated.

3.3 Improvement of the selected solutions

In this step, all the selected solutions are improved through a 2-opt algorithm. This algorithm deals with swapping facilities with equal sizes. In better words, only facilities which occupy the same number of free cells within the discretized laboratory area can be swapped. A swap is acceptable if it leads to a solution with better value for the objective function. All possible swaps are done until no more swap is possible.

To maintain the discipline adjacency constraint without checking all facilities neighbourhood for each swap, for each discipline, a graph of adjacency is maintained. Each facility of a discipline is a node of this graph and the adjacency between facilities is an edge between the two associated nodes. When switching facilities from different disciplines in the laboratory, the associated graphs are modified. Fulfilment of the adjacency constraint among the facilities of a similar discipline is verified through checking if the associated graph is still connected or not, once a swap is made.

4 Experimental results

To verify and validate the proposed heuristic approach, the layout of 46 facilities within a real laboratory with an area of about 300 square meter is designed. Figure 2 presents the area of the studied laboratory. In this figure, black rectangles show the forbidden or pre-occupied areas where no facility can be placed and the red square presents the entrance. Facilities lie into eleven disciplines with unequal sizes.

To apply the proposed heuristic approach, laboratory area and all facilities are discretized with the scale of one meter. Experimental results show that the proposed heuristic is able to provide diverse efficient solutions within a few seconds. The best solution found for this problem is presented in figure 3. This solution proposes a dense layout leaving free space around to allow decision makers to expand corridors and clearances with minimal loss in relative placement quality. As the solution presents, all facilities of a same discipline are neighbours which seals on the adjacency constraints fulfilment.

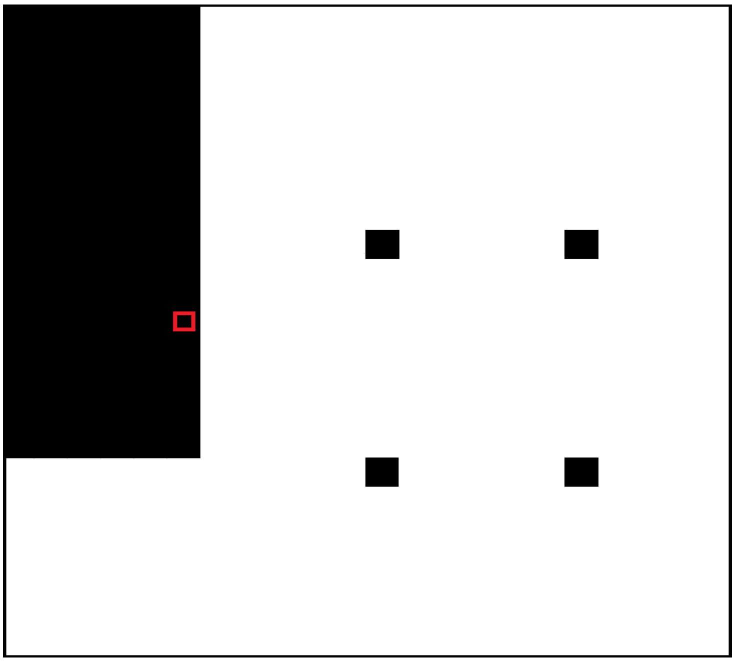


Figure 2. Area of the studied laboratory

5 Conclusion and perspectives

In this study, a heuristic approach in which a construction heuristic is integrated with a B&B method is developed to tackle facility layout problem in a standalone clinical laboratory. This approach aids to provide efficient and diverse solutions respecting physical, technical and organizational constraints. To evaluate the efficiency of the proposed approach, the layout of a real clinical laboratory is designed where 46 facilities must be placed within an area with pre-occupied places. The results present a feasible and a satisfactory design from the experts' point of views.

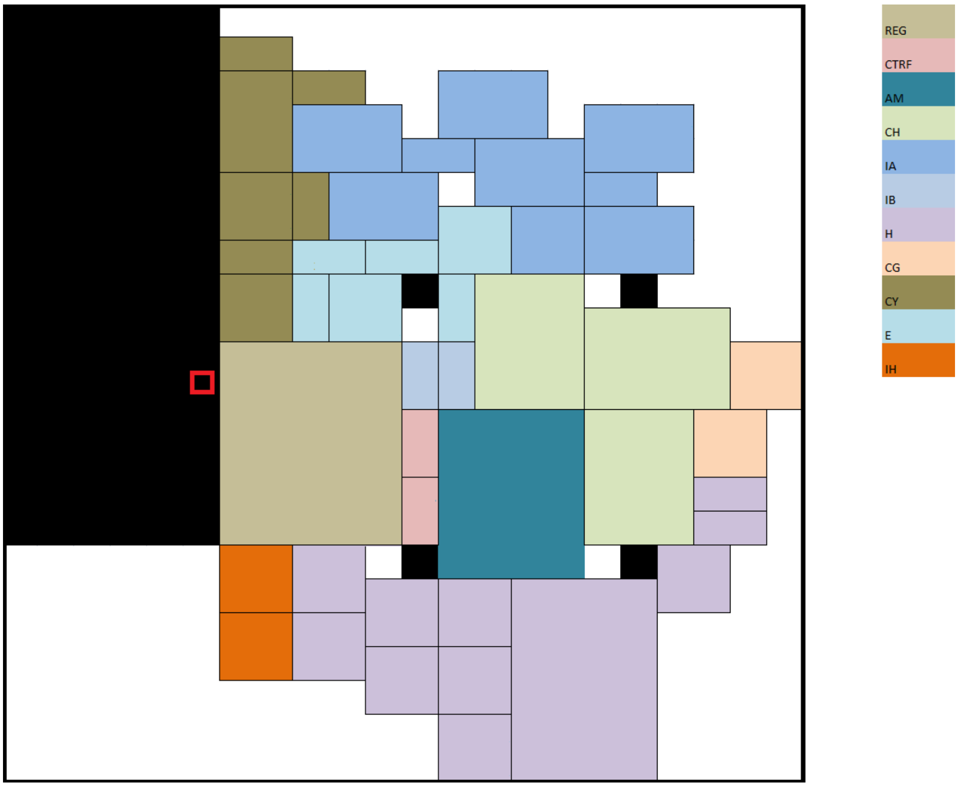


Figure 3. Best solution of the case study

Applying the proposed approach with different discretization scales is an interesting future research as in one hand, less discretization scale brings more details into design; but, on the other hand, it increases problem complexity and probably makes the problem expensive to be solved. Furthermore, as the proposed approach provides various number of efficient solutions in terms of total traveling distance, a computer simulation model is proposed to be applied in order to precisely evaluate the selected solutions in terms of other key performance indicators (KPIs) disregarded in this study. Such simulation model can be spotted as a complementary tool for decision making on the layout design.

References

[1] Hosseini-Nasab, H., Fereidouni, S., Ghomi, S. M. T. F., & Fakhrzad, M. B. (2018). Classification of facility layout problems: a review study. The International Journal of Advanced Manufacturing Technology, 94(1-4), 957-977.

[2] Koopmans, T. C., & Beckmann, M. (1957). Assignment problems and the location of economic activities. Econometrica: journal of the Econometric Society, 53-76.

[3] Heragu, S. S. (2008). Facilities design. CRC Press.

[4] Drira, A., Pierreval, H., & Hajri-Gabouj, S. (2007). Facility layout problems: A survey. Annual reviews in control, 31(2), 255-267.

[5] Singh, S. P., & Sharma, R. R. (2006). A review of different approaches to the facility layout problems. The International Journal of Advanced Manufacturing Technology, 30(5-6), 425-433.

[6] Samarghandi, H., & Eshghi, K. (2010). An efficient tabu algorithm for the single row facility layout problem. *European Journal of Operational Research*, *205*(1), 98-105.

[7] Nearchou, A. C. (2006). Meta-heuristics from nature for the loop layout design problem. International Journal of Production Economics, 101(2), 312-328.

[8] Pour, H. D., & Nosraty, M. (2006). Solving the facility and layout and location problem by ant-colony optimization-meta heuristic. International Journal of Production Research, 44(23), 5187-5196.

Optimizing injection blow molding by neuroevolution

H. Silva, R. Denysiuk, R. Pinto, F. Duarte and A. Gaspar-Cunha

IPC - Institute for Polymer and Composites, University of Minho,
Campus de Azurém, 4800-058 Guimarães, Portugal
agc@dep.uminho.pt

Keywords: injection blow moulding, multi-objective optimisation, evolutionary algorithms, neuroevolution.

Abstract

The industrial interest for the geometric optimization of blow moulded parts has been increasing over the last few years. Usually, the blow moulded part is divided into several sections, which have their thicknesses minimized thereafter. The quality of the optimized sections is limited by the shape of the container, and, because of that, sub optimal solutions are obtained. The present paper seeks to determine the optimal thickness distribution of blow moulded containers as function of geometry. The adopted methodology is based on the use of neural networks and finite element analysis. Neural networks are stochastically evolved, and they can consider several objectives at the same time, such as cost and quality. Numerical simulations are carried out using the commercial Finite Element Method (FEM) software. They are used to determine the mechanical behaviour of parts that underwent optimization processes. The methodology adopted in this work is applied to the design of an industrial bottle for applications such as the production of jars, bottles and similar containers. The ability of identifying critical zones regarding the mechanical behaviour of the part and of distributing material to minimize its worseness is present in the work. The developed methodology accounts varying thicknesses along the wall of the container. The results obtained in this work prove its validity and usefulness. The followed methodology is generic and can be applied to blow moulded containers with other geometries.

1 Introduction

Blow moulding is a widely used industrial process for the manufacturing of hollow plastic parts. The main applications of the blow moulding process are the production of jars, bottles and similar containers. These containers are widely used across the world, mainly to store beverages for human consumption, but also to contain cosmetics and oil. In the blow moulding process, a previously molten material is placed into a mould and inflated with gas, whose pressure forces the material to expand, allowing its shape to match that of the mould. The costs of raw materials represent a relevant fraction of the total costs of blow moulded products. Thus, the minimization of material usage leads to cost reduction and increases competitiveness of manufacturing companies. Achieve this goal requires a trade-off between production costs and quality criteria. In fact, reducing the amount of material may worsen significantly important product properties.

The common trial-and-error approach is tedious and inefficient when the aim is to optimize product development. It can lead to a significant waste of time and energy. Moreover, the originated results are very dependent on expert experience. The use of Computer Aided Engineering (CAE) has become increasingly popular to support engineering tasks. Computer simulations and optimization can help to reduce the number of empirical trials, thus saving time and money. Numerical approaches have a long history of use in blow moulding design. These approaches include Finite Element Methods (FEMs), gradient-based and stochastic search techniques, and neural networks.

The aim of this work is to apply multi-objective optimization using neuroevolution techniques, developed on a previous work, to optimize blow moulding designs. This methodology will be used to optimize the preform thickness profile of an industrial bottle before blowing.

This paper is structured as follows. The next section (section II) reviews existing works related to the optimization of blow moulding. Section III formulates the optimization problem addressed in this study. Section IV describes the developed algorithm. Section V presents and discusses the results of experimental study. Section VI concludes and outlines some possible future work.

2 Related work

Neural networks are often used to build a model that describes an underlying process from collected experimental data or to replace computationally expensive function evaluations during optimization. Some of the first studies about neural networks showed its ability to describe the blow moulding process with high accuracy. In the work [1], a neural network was used to predict the thickness distribution of the wall using the preform thickness distribution. The diameter and thickness swells of the preform in the blow moulding process based on extrusion were predicted from several operation parameters in [2]. Neural networks with a topology obtained empirically and trained by backpropagation were used to model a preform extrusion process [3]. The approach is based on experimental data. Two models were developed therein: one to predict the preform length at a certain time step based on temperature and extrusion flow rate, another to predict the thickness swelling of 20 points along the preform during its extrusion process for given parameters, such as temperature, extrusion flow rate and preform length. In [4], a Radial Basis Function (RBF) neural network was used with the aim of building a model for the internal and external temperature profiles of the preform related to power settings of infrared lamps in the heating oven.

Regarding the optimization of blow moulding process, two main problems can be identified. The first is to determine an optimized wall thickness distribution of the final container. The second concerns finding the shape of the preform with the aim of producing a container with a desired thickness distribution, defining appropriate process parameters. Optimization of thickness distribution is typically carried out by dividing the preform or container into several sections and optimizing the thickness of each section independently. In [5], the optimal preform thickness distribution that yields a given uniform part thickness was searched for. In the work [6], a performance optimization approach that aims to find the part thickness distribution that minimizes the mass of the part, while complying with mechanical constraints was developed.

Another problem which arose in blow moulding is concerned with process optimization, which aims at finding the optimal operating conditions for the minimization of the mass of the part and regarding the thickness distribution found by performance optimization.

In [7], the authors presented an approach to optimize the stretch blow moulding process. The main objectives were to establish the optimal preform geometry, in terms of thickness and shape and the optimal operating conditions to produce a container with a target thickness distribution. In the studies presented above, optimization was performed by gradient-based search methods. These methods present good theoretical properties and fast convergence as their main advantages. However, gradient-based methods are essentially local search techniques, which mean that their performance is highly dependent on the provided initial point.

Evolutionary algorithms (EAs) made possible to overcome limitations that are associated with traditional optimization methods. EAs are stochastic search methods that typically work by achieving a solution by processing a population. These algorithms allow performing global search without the need of using gradient information. In [8], a Genetic Algorithm was used to find the optimal thickness distribution for preform, regarding 25 positions distributed uniformly along the preform, with the aim of producing a blow moulded bottle with the required wall thickness distribution. A neural network was used to compute an approximation of the objective function implying to minimize the deviation from the target bottle distribution. This allowed obtaining the relationship between preform thickness distribution. The preform thickness distribution was built based on the results of FEM simulations for varying preform thickness distributions. In [4], a Particle Swarm Optimization (PSO) was used to adjust parameters of RBF network, to fit experimentally collected data. The next step was to apply an improved PSO method to obtain appropriate lamp settings and reducing energy waste. To obtain a target wall thickness distribution, the preform geometry was optimized in [9]. In [10], preform geometry optimization also includes the optimization of processing parameters. A common feature of the presented works that limits the applicability of the discussed approaches is that a container to be optimized is divided into several sections, and uniform thickness is defined within each section. A proper division may not be straightforward and is highly dependent on the container geometry. Poor results can be obtained if sections are unsuitably defined. Moreover, such approach can lead to discontinuities in joints between sections. The main contributions of this paper are the application of a regression model as a function of the container geometry to find the optimal thickness distribution and the solving the problem using a multi-objective neuroevolutionary algorithm.

3 Problem formulation

The aim of this study is to develop a methodology for the optimization of material usage in blow moulded objects. This topic is a major concern for industry, due to significant influence of costs of raw materials on the total production costs. The specific industrial bottle studied in this work has a diameter of 395 mm and a height of 625 mm. The material is PET with mass density of 1.15×10^{-9} kg/m3 and Poisson's ratio of 0.4.

Due to the complexity of the manufacturing process, the optimization was broken down into different steps. In the first step, given the geometry model of bottle, its optimal thickness profile was determined by optimizing the wall thickness distribution and mechanical properties [11]. This was performed by considering the bottle subjected to an inflation pressure of 2 MPa setting the minimum and maximum allowable values of the wall thickness to 1 and 5 mm, respectively. The second step is considered in the present study. It consists of determining the optimal thickness profile of the preform to the bottle with the desired thickness profile is obtained as the result of the preform inflation in the blow moulding process.

Figure 1 shows the process to be simulated and optimized. The figure outlines major steps in the injection blow moulding. First, the molten polymer is injected into a heated preform cavity that is clamped around a blowing rod that forms the internal shape of the preform. Then, air at a certain pressure is introduced inside the heated preform to force the material to acquire the internal shape of the mould, which corresponds to the desired shape of the bottle. As the optimal thickness profile of bottle was determined, the goal of this study is to find the optimal thickness profile of the preform that will produce the desired bottle because of the process shown in Figure 1. It might be possible to note that the optimization goes in the opposite direction to the steps of blow moulding process.

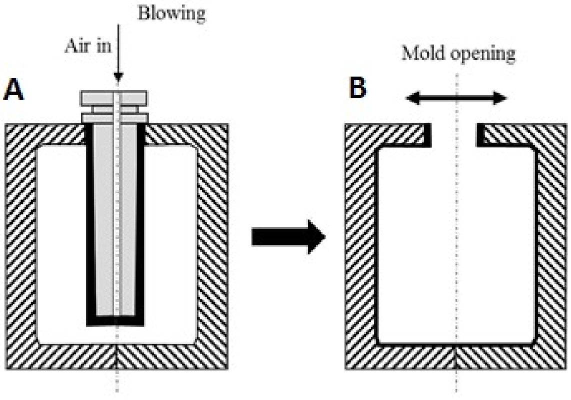


Fig. 1. The injection blow moulding process.

Figure 2 shows the geometry model of the preform and mould used in this study. The problem consists of determining the wall thickness distribution of preform to the injection blow moulding shown in Figure 1 will results in the bottle with the given thickness profile. Using the finite element model, we address this problem by multi-objective optimization. The variables to be optimized are the thickness values in the nodes of the FEM of preform while the objectives correspond to the deviation from the desirable thickness values in the nodes of the bottle mesh.

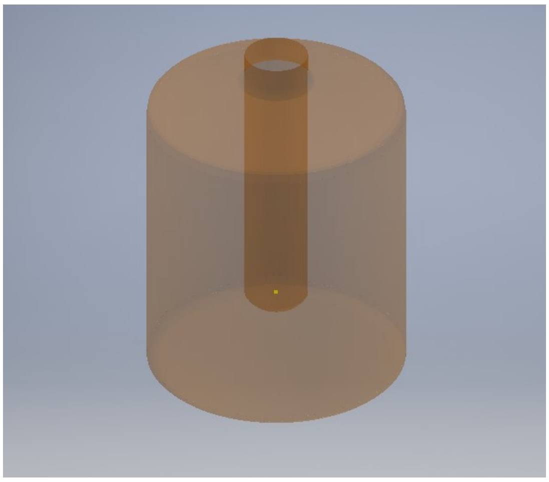


Fig. 2. Geometry model of preform to be inflated and mould.

The objectives to be minimized can be formally defined by equations (1) and (2).

$$f_1 = \frac{1}{M}\sum_{i=1}^{M}\frac{|t_i^0 - t_i^1|}{t_i^0} \qquad (1)$$

$$f_2 = \max_{1 \leq i \leq M}\frac{|t_i^0 - t_i^1|}{t_i^0} \qquad (2)$$

where M is the number of nodes in FEM of bottle, for $i = 1, \ldots, M$, t_i^0 is the value in the i-the node of the target thickness distribution that was obtained on the previously optimization works, which consists in the best final parts profile taking account the maximization of mechanical behavior and minimization of final weight, and t_i^1 is the thickness value resulting from using injected blow molding with a given thickness profile of preform. These objectives are calculated by collecting results from computer simulations performed using ANSYS Workbench, the commercial Finite Element Analysis software.

4 Optimization methodology

This section details on the methodology proposed for designing blow moulded containers with optimized wall thickness distribution. The main idea is to consider the wall thickness as a function of the container's geometry. The achievement of this consideration depends on the capabilities of neural networks. Neural networks are used for the conversion of the coordinates along the wall into thickness values. This can also be considered as a regression model. Despite that, it is important to point out the difference between traditional regression and the proposed methodology. The traditional regression makes use of data points with known input and target values. In the proposed methodology, only the values of input variables are available, which limits the applicability of traditional gradient-based methods to learn the parameters of the neural network.

To achieve the desired behaviour, the parameters of neural network must be adjusted. As the applicability of traditional methods based on gradients is limited, neuroevolution approach is adopted in this work. Neuroevolution refers to the use of evolutionary algorithms to evolve neural networks [12]. Although neuroevolution provides the potential to evolve both the parameters and the topology of neural networks, in this study we keep the topology fixed when evolving the parameters (weights and biases).

The main steps of the approach are given by Algorithm 1. It is based on SMS-EMOA [13], which in its turn is a popular state-of-the-art algorithm for evolutionary multi-objective optimization that proved effective in several studies and real-world application. The algorithm starts by initializing a population of individuals. Each individual in the population is represented by a real-valued chromosome that encodes the parameters (weights and biases) of a neural network. Each time a new individual is generated it is sent for evaluation. The evaluation procedure comprises decoding the individual's chromosome into the neural network, calculating the thickness profile of the preform and computing the values of objective functions which reflect the individual's performance.

The notion of the thickness profile refers to the values of wall thickness in any location of the preform. Figure 3 graphically illustrates the idea behind the calculation of the thickness profile of the preform. The coordinates of each mesh node in the finite element model are fed into the neural network. The output is the thickness at the corresponding nodal location. This way, propagating the coordinates of all the mesh nodes through the neural network originates a thickness profile of the preform.

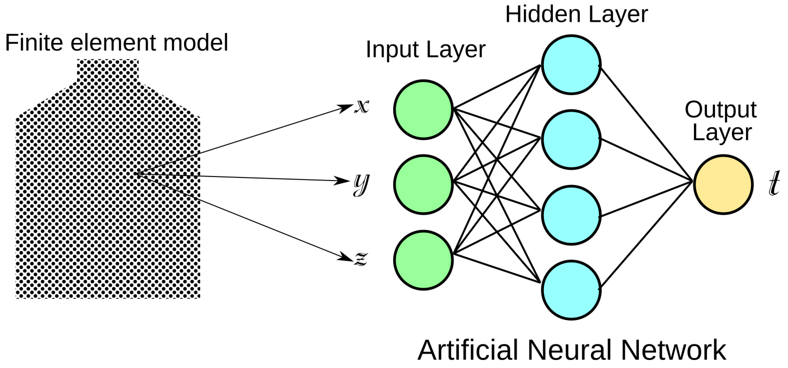


Fig. 3. Thickness profile calculation.

The initialization procedure is followed by a steady-state variant of the evolutionary process (lines 11-22), meaning a single offspring is produced in each generation, that is performed to evolve the initial population of solutions. It encompasses three major steps: selection, variation and replacement. Selection aims at selecting a pool of promising parent individuals (R) for producing offspring. This study uses a simple uniform selection, where each population member has equal chance to be selected. Evolutionary operators are then applied to the parents, to produce offspring in the variation procedure (lines 15-18) where a new offspring is generated, evaluated and added to the population.

Variation plays a crucial role in the exploration of the search space. As a real representation of the chromosome is used in this study, a Differential Evolution (DE) operator is employed for producing offspring. It is a simple yet effective evolutionary operator. DE generates a new solution q by exploiting the differences between solution vectors in the population. It can be described by equation (3).

$$y_j = \begin{cases} x_j^{(1)} + F(x_j^{(2)} - x_j^{(3)}) & \text{if } r_j \leq CR \\ x_j^{(1)} & \text{otherwise} \end{cases} \qquad (3)$$

for $j = 1, \dots, n$, where CR is the probability of crossover and F is the scale factor and $r_j \sim \mathcal{U}(0,1)$. Thereafter, a Polynomial Mutation (PM) is applied to introduce additional variations to ensure the exploration of new regions of the search space. This operator gives a higher probability that a new solution is closer to rather than far away from the previous solution. It can be defined by equation (4).

$$y_j = \begin{cases} y_j + \delta_j(ub_j - lb_j) & \text{if } r_j \leq p_m \\ y_i & \text{otherwise} \end{cases} \qquad (4)$$

with the value of δ_j being given by equation (5).

$$\delta_j = \begin{cases} (2u_j)^{1/(\eta_m+1)} - 1 & \text{if } u_j \leq 0.5 \\ 1 - (2 - 2u_j)^{1/(\eta_m+1)} & \text{otherwise} \end{cases} \qquad (5)$$

For j=1,…, n, where lb_j and ub_j are the lower and upper bounds, rj, uj ~ $\mathcal{U}$(0,1), whereas $p_m \in$[0,1] and η_m>0 are control parameters (a probability of mutation and a distribution index, respectively).

Algorithm 1 Neuroevolution

```
1:  Input: $\mu$, $t_{max}$
2:  // initialization
3:  $t \leftarrow 0$
4:  $P \leftarrow \{\}$
5:  for $i \in \{1, \ldots, \mu\}$
6:      $t \leftarrow t + 1$
7:      $p \leftarrow Initialize\,()$
8:      $p \leftarrow Evaluate\,(p)$
9:      $P \leftarrow P \cup p$
10: end
11: while $t < t_{max}$
12:     $t \leftarrow t + 1$
13:     // selection
14:     $R \leftarrow Select\,(P)$
15:     // variation
16:     $y \leftarrow Recombine\,(R)$
17:     $y \leftarrow Evaluate\,(y)$
18:     $P \leftarrow P \cup y$
19:     // replacement
20:     $\mathcal{F}_l \leftarrow Sort(P)$
21:     $P \leftarrow P \backslash \mathrm{argmin}_{a \in \mathcal{F}_l} \mathcal{S}(\mathcal{F}_l) - \mathcal{S}(\mathcal{F}_l \backslash a)$
22: end
23: Output: $P$
```

The replacement procedure (lines 19-21) aims at forming a population of the next generation by removing a worst performing individual thereby keeping the population size equal to the predefined value. In doing so, it relies on the concept of survival of the fittest from natural evolution. Because the proposed neuroevolution is designed to deal with multiple objectives, this procedure must ensure the convergence and diversity of population. These two requirements are known to be somewhat conflicting in nature. The adopted replacement strategy relies on the concept of Pareto dominance to provide convergence and the hypervolume measure to ensure diversity. In the beginning, the population is sorted using the non-dominated sorting procedure so as to find individuals in the last non-dominated front ($\mathcal{F}_l$). Then, the one with the smallest hypervolume contribution in $\mathcal{F}_l$ is removed. This relies on the notion of hypervolume. For a set of objective vectors in $\mathcal{F}_l$, the hypervolume can be defined as the Lebesgue measure Λ (or $\mathcal{S}$ measure) of the set of objective vectors that are dominated by $\mathcal{F}_l$ but not by a reference point (r), i.e.,

$$\mathcal{S}(\mathcal{F}_l) = \Lambda\left(\cup_{a \in \mathcal{F}_l} \{y | a \prec y \prec r\}\right). \qquad (6)$$

In this study, when computing the hypervolume, we use the normalized objective values that are in the range [0,1] the reference point of $r = 1$. The result of the process described above is expected to be a set of neural networks representing different trade-offs between the defined objectives, with each neural network giving the design of preform in terms of the wall thickness distribution.

The implementation of the proposed methodology makes use of the two commercial software: ANSYS and MATLAB. An illustration of the overall software interaction is shown in Figure 4. The optimization program that is outlined in Algorithm 1 is implemented in MATLAB. It aims at finding the parameters of the neural network that produces the thickness profile of the preform so as to optimize the objectives defined in (1) and (2). These objectives includes both the thickness profiles of the preform and bottle. The latter is obtained by a computer simulation performed using ANSYS, the Finite Element Analysis software. ANSYS workbench is run in a batch mode, as a result of being invoked by the MATLAB optimization routine each time a new solution needs to be evaluated. The communication between the two software programs is established through text files.

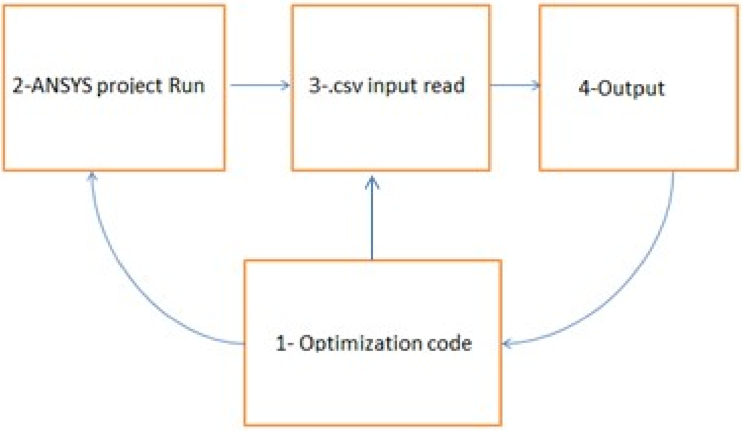
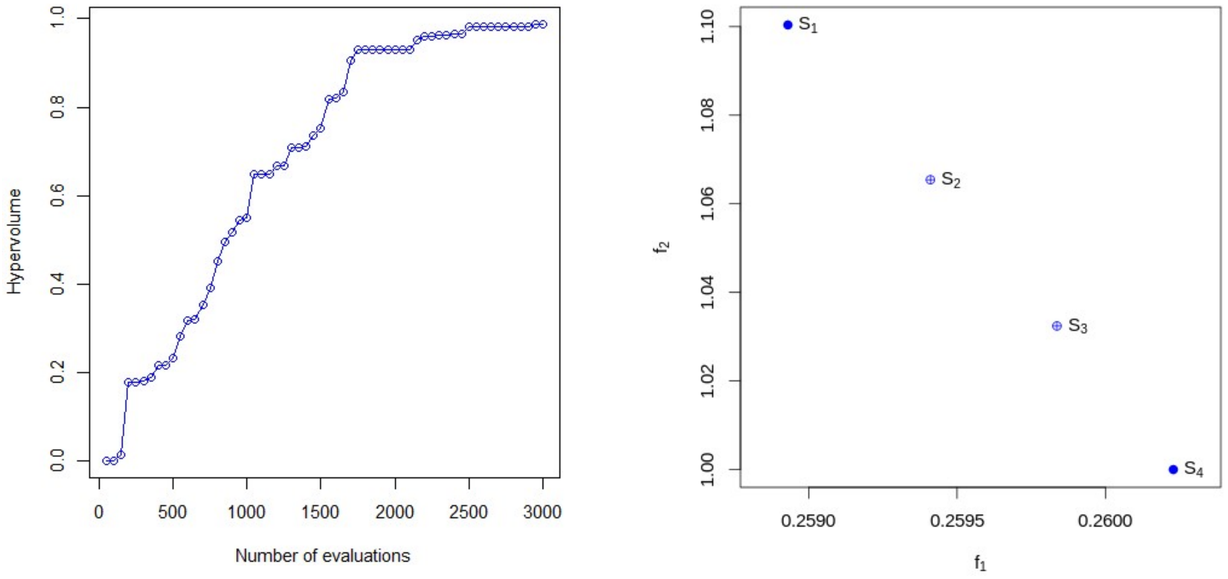


Fig. 4 - Flow chart of the optimization methodology.

5 Computational experiments

The numerical simulations based on finite element analysis were carried out by ANSYS workbench. Due to high a computation cost, the optimization was performed under a limited computational budget. The neuroevolution algorithm was run for 3000 function evaluations with the population of 50 individuals, the crossover probability of CR=1, the scaling factor of F=0.5, the probability of mutation of p_m=1/n (where n is the chromosome length) and the distribution index of η_m=20.

Figure 5-A) depicts the evolution of the hypervolume throughout the generations. The proposed neuroevolution gradually drives the population of solutions to better regions of the search space improving the thickness profile of the preform in terms of the defined objectives. It also can be seen that a major improvement takes place in early generations whereas the later generations are mostly spent for fine-turning. This is a common feature of evolutionary search, and this result suggest that integrating a local search procedure is a valid way for future improvement. Figure 5-B) shows the set of obtained nondominated solutions. The obtained solutions represent different trade-offs between the objectives. Thus, the solution with the minimum average error is found for a larger deviation in the individual node, whereas the solution with smaller maximum error corresponds to a larger mean error. The two solutions on the extremes of the trade-off curve are highlighted and used for further analysis.



A) B)

Fig. 5. A) Evolution of the hypervolume; B) Nondominated solutions after 60 generations.

In Figure 6 it is possible to see graphical Finite Element results, taken from ANSYS simulation. The global thickness results vary from 1.60×10^{-4}m to 2.227×10^{-3} m. However, the side walls of the model present values within the range of 1.00×10^{-3} m approximately.

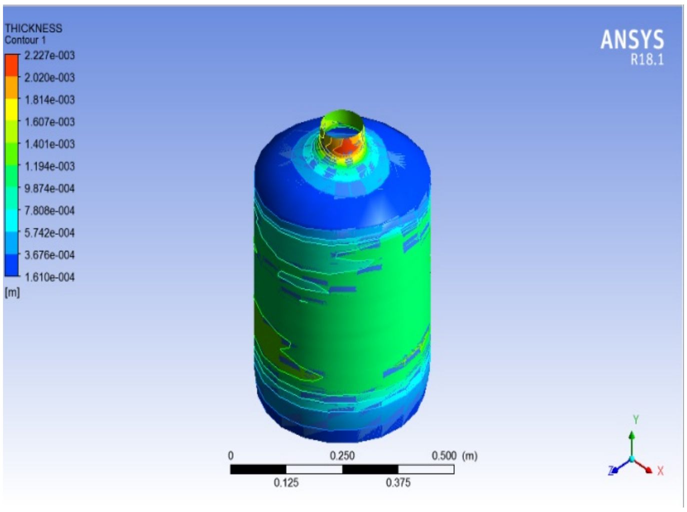
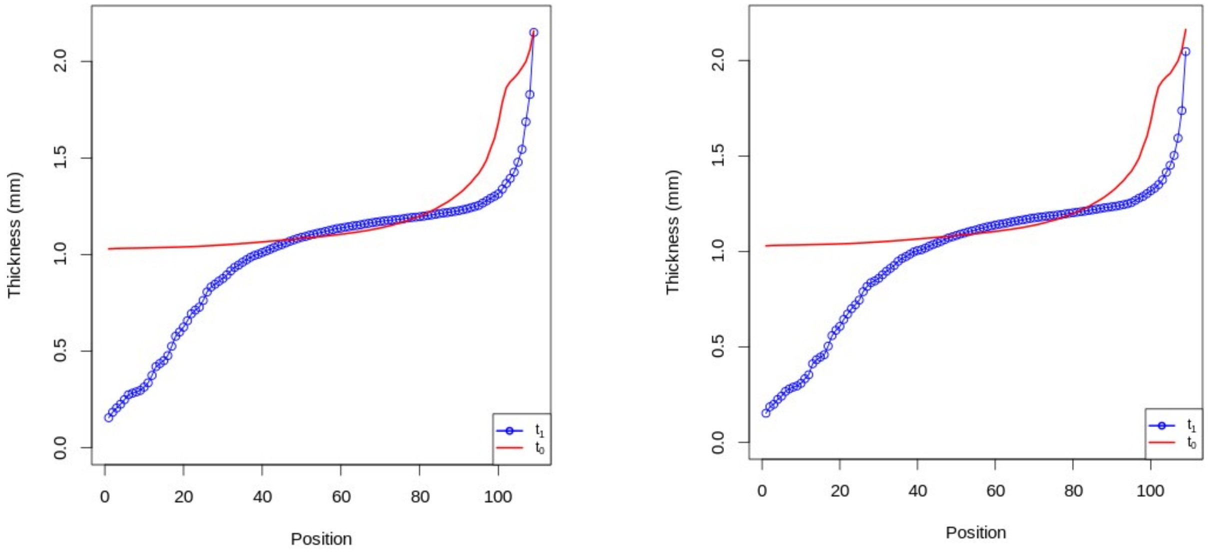


Fig. 6. Graphical thickness results for the solution S_1.

The thickness distribution of the FEM models was obtained for both the preform (Figure 7) and the bottle (Figure 8). Results were collected throughout 110 nodal points from the walls of the model along a vertical line. The y coordinate (vertical axis) has the maximum value at the top end (bottleneck), and the lowest value at the bottom of the model. The position number increases with decreasing y coordinate, i.e, in the direction from the top to the bottom of the model. Same nodal points were used for both the preform and the bottle model, although their coordinates change during inflation.



A) B)

Fig. 7. A) Thickness distribution of the bottle for S_1; B) Thickness distribution of the bottle for S_4.

Figures 7 presents the thickness distribution for both solutions (S_1 and S_4), and the target solution obtained in previous work (t_0), is close to 1.0 for positions from the beginning until position 80, although slightly increasing. The thickness t_0 increases sharply thereafter. Both thickness distributions obtained in the present work shows similar behavior to t_0 after 40th position, while from positions 0 to 50 presents some discrepancy. This can be explained by the fact that the previous work used a uniform internal (static) pressure as loading, while the present work is considers the inflation of the preform mold. As the mold is inflated, the air blows from the bottleneck until the bottom of the preform. As the bottleneck has no material above, the thicknesses close to the free end will behave differently as the same ones in the t_0. For positions higher than 50, there is a better agreement between the two distributions.

Figure 8 shows the wall thickness distribution of the preform. Both the solution S_1 and the solution S_4 represent a similar behavior and show very close values overall. Both S_1 and S_4 show a strict increase of the thickness with increasing y position. This reveals the ability of the optimization to distribute the material according to the established requirements. A small difference between the thickness profiles of the preform represented by these solutions may indicate only a slight conflict between the objectives.

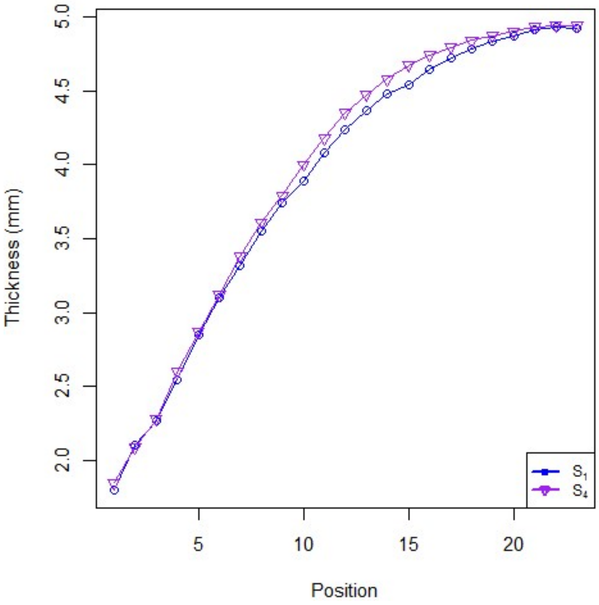


Fig. 8. Thickness distribution of the preform for S_1 and S_4

6 Conclusions

In blow moulding industry, the product competitiveness can be effectively increased by reducing the costs of raw materials. This study suggested a methodology to optimize the material usage in blow moulded products. This methodology aims at determining the optimal distribution of material as a function of the product geometry. Motivated by the universal approximation property, this function is approximated by neural network. The structure and parameters of the network are determined by neuroevolution. The search is performed addressing multiple objectives, minimizing the usage of material and the degradation of mechanical properties. This leads to a set of Pareto optimal networks representing different trade-offs between the objectives, which allows acquire a valuable information about design alternatives and enables a posteriori decision making.

The application of the proposed methodology is demonstrated in a case study addressing the design of industrial bottle. Finite element analysis software was employed to simulate the response of the specific design to a static pressure. The obtained results indicate the importance of using proper search strategies and the ability of neuroevolution to optimize the thickness distribution under given conditions. Generality is a major advantage of the proposed methodology, as its applicability is independent of the bottle geometry.

In future, the developed methodology will be extended to include the operating conditions and the physical properties of the material into optimization.

Aknowledgements

This work has been supported by FCT - Fundação para a Ciência e Tecnologia in the scope of the project: PEst-OE/EEI/UI0319/2014 and the European project MSCA-RISE-2015, NEWEX, with reference 734205.

References

[1] R. W. Diraddo and A. Garcia-Rejon (1993). On-line prediction of final part dimensions in blow molding: A neural network computing approach. Polym. Eng. Sci., 33(11), 653-664.

[2] H. X. Huang and C. M. Liao (2202). Prediction of preform swell during extrusion blow molding using neural network method. Polym. Test., 21(8), 745-749.

[3] H. X. Huang and S. Lu (2005). Neural modeling of preform extrusion in extrusion blow molding. J. Reinf. Plast. Compos., 20(10), 1025-1034.

[4] Z. Yang, W. Naeem, G. Menary, J. Deng and K. Li (2014). Advanced modelling and optimization of infared oven in injection stretch blow-moulding for energy saving. in Proceedings of the 19th IFAC World Congress, Cape Town.

[5] D. Laroche, K. K. Kabanemi, L. Pecora and R. W. DiRaddo (1999). Integrated numerical modeling of the blow molding process- Polym. Eng. Sci., 39(11), 1223-1233.

[6] C. Gauvin, F. Thibault and D. Laroche (2003). Optimization of blow molded part performance through process simulation. Polym. Eng. Sci.,. 43(12), 1407-1414.

[7] F. Thibault, A. Malo, B. Lanctot and R. Diraddo (2007). Preform shape and operating condition optimization for the stretch blow molding process. Polym. Eng. Sci., 47(3), 289-301.

[8] G. Q. Huang and H. X. Huang (2007). Optimizing preform thickness for extrusion blow molding by hybrid method. J. Mater. Process. Technol., 182(8), 512-518.

[9] J. Biglione, Y. Béreaux, J. Y. Charmeau, J. Balcaen and S. Chhay (2016). Numerical simulation and optimization of the injection blow molding of polypropylene bottles - A single stage process. Int. J. Mater. Form., 471-487.

[10] C. Hopmann, S. Rasche and C. Windeck (2015). Simulative design and process optimization of the two-stage stretch-blow molding process. in AIP Conf. Proc..

[11] R. Denysiuk, F. M. Duarte, J. P. Nunes and A. Gaspar-Cunha (2017). Evolving Neural Networks to Optimize Material Usage in Blow Molded Containers. in EUROGEN, Madrid.

[12] D. Floreano, P. Dürr and C. Mattiussi (2008). Neuroevolution: from architectures to learning. Evol. Intell., 1(1), 47-62.

[13] N. Beumea, B. Naujoksa and M. Emmerich (2007). SMS-EMOA: Multiobjective selection based on dominated hypervolume. Eur. J. Oper. Res., 181(3), 1653-1669.

Reducing environmental impacts in heat exchanger networks using Life Cycle Assessment and metaheuristic optimization techniques

Leandro V. Pavão[1], Caliane B. B. Costa[1] and Mauro A. S. S. Ravagnani[1]

1. Department of Chemical Engineering, State University of Maringá, Av. Colombo, 5790, Bloco D90, CEP 87020900, Maringá, PR, Brazil
leandropavao@gmail.com; cbbcosta@uem.br; massravagnani@uem.br

Abstract. Energy integration via heat exchanger networks (HEN) aiming for capital and operating costs minimization has been one of the most widely studied fields in the scope of process engineering. It is evident that heat recovering in industrial plants yields pollutant emissions mitigation. However, studies performing direct assessment on of the environmental impacts (EI) that HEN synthesis aids on reducing are relatively recent and therefore still scarce. This work aims to contribute to the area by applying a new superstructure-based optimization model for HEN synthesis entailing both economic and environmental performances. In the new methodology, EI were assessed via Eco-costs, a strategy based on Life Cycle Assessment (LCA) and that translates EI into monetary values, rendering a single-objective model. The model considers different utilities and fuels and the environmental impacts they entail as well as the impacts originated from heat exchangers construction. The solution strategy used is based on a two-level hybrid meta-heuristic method, which uses a combinatorial optimization method (Simulated Annealing) for optimizing binary variables and a continuous optimization approach (Rocket Fireworks Optimization, a combination of Continuous Simulated Annealing and Particle Swarm Optimization) for the continuous variables. An industrial-sized case study was used with utilities produced using different energy sources. The problem was solved under four different conditions. In Case 1, only HEN-costs (HENC) are considered. In Case 2, only Eco-costs (EC) are considered. In Case 3, EC and HENC were considered as objective function. Case 4 assumed a hypothetical scenario where EC was limited to a maximum value by a local environmental policy, so that a solution with minimal HENC had to be found constrained to such EC upper bound. Less environmentally damaging utility sources such as those using natural gas as fuel or cooling towers were favored in situations where EC were included. Moreover, the method was able to successfully handle the upper bound constraint in Case 4.

Keywords : optimization, heat exchanger networks, metaheuristics, environmental impacts, eco-costs, life cycle assessment

1 Introduction

Different fluid streams in industrial processes might require heating or cooling. Such temperature manipulation tasks are carried out in heat exchangers (HE). A HE can transfer heat from a hot stream to a cold one. If they are process streams, the demand for hot and cold utilities (*e.g.*, steam and cooling water) in the process is mitigated. A set of HE performing heat integration in a plant is called a heat exchanger network (HEN). Given that industrial plants might encompass several streams requiring heating/cooling, the optimal HE allocation is a challenging task. HEN synthesis is among the most studied fields in process engineering, consisting of a mathematically complex problem both in the combinatorial and continuous domains.

Different mathematical models based on superstructures have been developed and applied to a variety of case studies. One of the most important contributions for HEN synthesis based on mathematical programming is the stage-wise superstructure (SWS) [1]. That superstructure served as basis to other stage-wise models such as the no-splits model [2] and the hybrid splits/no-splits model [3]. The model is relatively simple, comprising all possible stream match combinations at all stages. Stream splits are possible, and each match is placed in a different stream branch. Heaters and coolers are placed at the end of cold and hot streams, respectively, providing auxiliary temperature manipulation.

Several solution approaches have been used in the literature to identify optimal solutions for the HEN synthesis problem. Recently, metaheuristics have arisen as a strong tendency for achieving promising solutions to the problem, being applied in several works. Some important advantages of such sort of method are the relative ease of implementation, the possibility of using coding platforms that are free of charges and the simplicity in handling disjunctions and thermodynamic constraints for maintaining solutions feasible. Important works under that scope are worth mentioning. Ravagnani *et al.* [4] employed Genetic Algorithms (GA) with Pinch Analysis concepts. Silva *et al.* [5] used Particle Swarm Optimization (PSO) applied to the SWS model. Peng and Cui [2] solved a no-splits model with two-level Simulated Annealing. A hybrid GA-PSO with parallel processing was employed to the SWS model by Pavão *et al.* [6], who later presented a hybrid SA-PSO method applied to a no-splits model [7] and hybrid SA with Rocket Fireworks Optimization (RFO, a combination of Continuous SA with PSO) used to solve the SWS [8]. Random Walk Algorithm with Compulsory Evolution was applied to the SWS model by Bao *et al.* [9] and Cuckoo Search Algorithm was employed to a no-splits model by Zhang and Cui [10].

For long, the discussions on the HEN synthesis field have had, on their majority, a strong economic bias. However, with the ever-increasing concerns on pollution and global warming, the debates began pointing towards the attainment of cleaner production processes. On that trend, Life Cycle Assessment (LCA) techniques arose as important methodologies for environmental impacts evaluation and quantification. Those concepts have been considered only recently in HEN synthesis works. López-Maldonado *et al.* [11] presented multi-objective optimization framework considering environmental impacts and total annual costs (TAC). Those authors used the Eco-indicator 99 approach to quantify the environmental impacts associated to their solutions. The model used was based on the SWS with one additional branch per stage containing a heater/cooler. They used deterministic solvers in GAMS platform. Vaskan *et al.* [12] also used Eco-indicator 99, but performed analyses to different categories of EI with a dimensionality reduction technique. Those authors employed the SWS model and solved their models with deterministic solvers in GAMS. Ravagnani *et al.* [13] were the first to employ a metaheuristic approach for analyzing HEN regarding both the financial and environmental performances. The latter was quantified by the authors with SimaPro software. The MOO technique used was a modified PSO strategy. Pavão *et al.* [14] used a modified version of SA-RFO [8] to solve a SWS-based model. The ReCiPe approach was employed for the environmental impacts quantification. Mano et al. [15] used the Eco-costs technique, which translates EI into monetary values that can be summed to the HEN costs, entailing thus a single-objective optimization model (which the authors solved with deterministic techniques in GAMS environment). The eco-costs are calculated based on hypothetical money amounts needed to prevent a given environmental impact. A database of such costs for several industrial processes, as well as the methodology related information can be found in the website maintained by the Delft University of Technology [16].

According to Furman and Sahinidis [17], the HEN synthesis problem is NP-hard in the strong sense. Thus, considering both financial and environmental performances makes the problem even more laborious. Given such difficulties, HEN synthesis works that consider financial and environmental performances with deterministic techniques consider case studies with a small number of streams (*e.g.*, three streams [12]; seven streams [11]). Pavão *et al.* [14], with their MOO metaheuristic, were able to approach larger cases (up to 15 streams). However, the SWS version used there could handle only one type of hot and one type of cold utility. Given the potential of SA-RFO to handle large-scale HEN synthesis cases, in this work a large-scale (20 streams) case study from the literature is adapted to comprise different utility types that are produced from different sources: two boilers burning different fuels producing steam at three different pressures and one hot oil supplier are responsible for auxiliary heating, while cooling towers, chillers and compressors are available to provide cooling water, cold water and compressed air for cooling tasks. The Eco-costs approach [16] used by Mano *et al.* [15] proved to be efficient, and the case study environmental performance is analyzed here under that approach, avoiding the need for a multi-objective formulation.

2 Formulation

The stage-wise superstructure model used here comprises additional stream split branches for each type of utility, at all stages, as illustrated in Figure 1. This figure also depicts that utilities may be produced using different sources, entailing different environmental impact values. These values depend, for instance, on the fuel or electricity source, and on a utility producing unit’s efficiency.

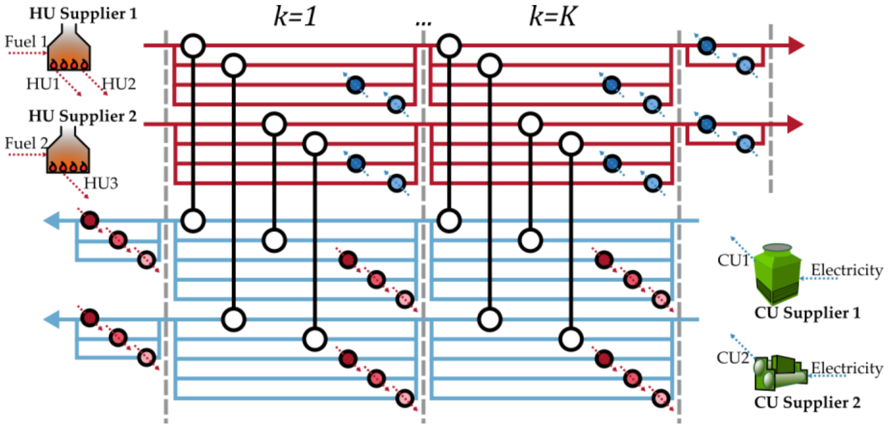


Figure 1. Stage-wise superstructure with multiple utilities

The objective function is a sum of monetary values: capital and operating costs and Eco-costs related to utilities and to HE construction. The utilities-related eco-costs (UEC) are calculated as follows:

$$
\begin{aligned}
UEC = \; & \sum_i \sum_n ECcu_n \cdot FhQcuEnd_{i,n} \cdot QTcuEnd_i + \\
& \sum_i \sum_n \sum_k ECcu_n \cdot QcuInt_{i,n,k} + \\
& \sum_m \sum_j EChu_m \cdot FcQhuEnd_{m,j} \cdot QThuEnd_j + \\
& \sum_m \sum_j \sum_k EChu_m \cdot QhuInt_{m,j,k} + \\
& i \in N_H, m \in N_{HU}, j \in N_C, n \in N_{CU}, k \in N_S
\end{aligned}
\qquad (1)
$$

where the *cu* and *hu* suffixes stand for cold and hot utilities, the *EC* prefix stands for eco-costs (*e.g.*, the *EChu* variable are the eco-costs per hot utilities requirement in kW), the *Int* suffix stands for intermediate stages, *Q* prefix regards heat loads (*e.g.*, *QcuInt* is the heat load of a cooler at an intermediate stage), *QTcuEnd* and *QThuEnd* are total cold and hot utilities required in end stages and *FhQcuEnd* and *FcQhuEnd* are fractions of that total that are provided by each of the available utility types. The indexes *m* and *n* are related to hot and cold utility types.

Heat exchangers construction eco-costs (HECEC) are calculated as follows:

$$
\begin{aligned}
HECEC = \; & ECA \times \left(\begin{aligned}
& \sum_i \sum_j \sum_k z_{i,j,k} \cdot A_{i,j,k} + \\
& \sum_i \sum_n \sum_k zcuInt_{i,n,k} \cdot AcuInt_{i,n,k} + \\
& \sum_m \sum_j \sum_k zhuInt_{m,j,k} \cdot AhuInt_{m,j,k} + \\
& \sum_i \sum_n zcuOut_{i,n} \cdot AcuEnd_{i,n} + \\
& \sum_m \sum_j zhuOut_{m,j} \cdot AhuEnd_{m,j} +
\end{aligned} \right) \\
& i \in N_H, m \in N_{HU}, j \in N_C, n \in N_{CU}, k \in N_S
\end{aligned}
\qquad (2)
$$

where ECA are eco-costs per square meter of heat exchangers area, the *A* prefix stands for areas, and *z* for the existence/absence of a given unit. Suffixes are analogous to those previously presented for Eq. (1).

The single-objective optimization model is written as minimization of total annual costs (TAC):

$$\begin{aligned} (HEN\_OPT) \quad & \min \quad \{TAC = OC + CC + UEC + HECEC\} \\ & s.t. \qquad\qquad HEN \quad constraints \end{aligned} \tag{3}$$

where OC and CC are yearly operating and capital costs, and their calculation by means of energy balances, design equations and feasibility constraints has been extensively detailed for analogous models in the literature [14].

3 Methodology

The SA-RFO [14] strategy is a two-level HEN synthesis hybrid method. In the “outer” level of the method, SA is used to propose new topologies, while RFO optimizes continuous variables associated to that topology in a mutual feedback scheme, illustrated in Figure 2.

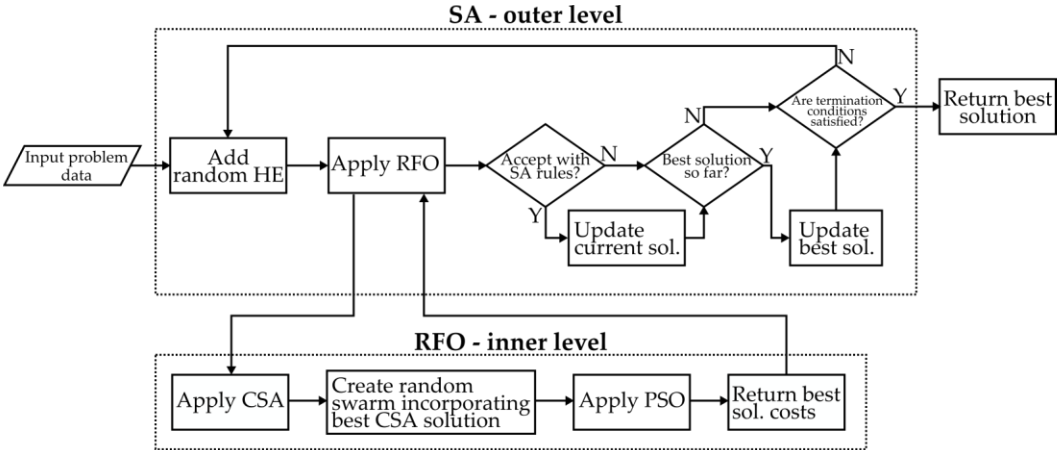


Figure 2. SA-RFO block diagram

Note that RFO is in fact a combination of two methods: a continuous SA and PSO. The former is likely able to achieve a promising solution. That solution is then incorporated to the PSO initial swarm, which better explores that region and may find better solutions.

4 Numerical example

A case study is proposed here to test the methodology presented. It has 10 hot and 10 cold streams, and is based on an example proposed by Luo *et al.* [18]. Its original stream and economic data can be found therein. The version found in Ref. [18] has one type of hot and one type of cold utility. In order to provide a more realistic scenario for environmental performance analysis, a hypothetical set of utilities that can be commonly found in industrial plants is proposed. Two boilers are available: the first burning coal and producing high and medium pressure steam and the second burning natural gas and producing medium and low-pressure steam. For cooling, three types of utilities are available: water from cooling towers, water from chillers and compressed air. The energy source in those cases is electricity. Table 1 presents the new utilities, their costs and the associated eco-costs. In the table header, *OT* is the operating temperature range and *h* is the heat transfer coefficient. All eco-cost values were obtained by means of extrapolation from values contained in the eco-costs online database [16].

Table 1. Case study utilities data

Utility, unit, source	Label	*OT* (°C)	*h* (kW/m^2°C)	Cost ($/kW)	Eco-cost ($/kW)
HP steam, boiler 1, coal	HU1	265	2.0	120	292.88
MP steam, boiler 1, coal	HU2	200	2.0	80	275.65
MP steam, boiler 2, natural gas	HU3	200	2.0	100	1135.45
LP steam, boiler 2, natural gas	HU4	150	2.0	50	1059.75
Cooling water, cooling tower, electricity	CU1	30-40	1.5	10	9.89
Compressed air, compressor, electricity	CU2	30-40	1.5	50	993.83
Cold water, chiller, electricity	CU3	5-10	1.0	5	98.91
Heat exchangers construction eco-cost: 7.91 $/m^2					

This application was divided in four cases:

(i) Case 1: only HEN costs are considered in the objective function during optimization, disregarding environmental impacts.

(ii) Case 2: only environmental impacts (*i.e.*, eco-costs) are considered in the objective function.

(iii) Case 3: a summation of costs and eco-costs is considered. In that case, the optimization is supposed to yield solutions with a reasonable trade-off regarding financial and environmental performance according to the eco-costs methodology.

(iv) Case 4: a hypothetical situation where, due to an environmental policy, the HEN eco-costs cannot be greater than a given value. Such value is assumed as 4.5 million dollars per year. Thus, the optimization must find a solution with minimal HEN costs for a pre-fixed value for eco-costs.

In Case 4, an additional upper bound constraint is included in the model. Solving the model with such constraint activated required additional coding to the SA-RFO method. The initial solution used in SA-RFO runs is a “null” solution, *i.e.*, no heat exchangers are present and heating and cooling are performed entirely via utilities at streams’ ends. When a topology is proposed by SA, the initial solution used in RFO is null as well: all heat exchangers present in that topology have zero heat loads, and all the temperature manipulations are carried out with utilities. It is thus likely that a null solution does not satisfy the upper bound constraint defined by the aforementioned environmental policy. A Boolean variable (*UBvalid*) was associated to the solutions (the current solution in SA or CSA, or a particle in PSO) containing information on whether such solution had at least once satisfied the eco-cost upper bound. If such condition is false, the optimization is carried out considering eco-costs only. If it is true, costs are set as the main objective. However, during the method iterations if a new solution assumes a value that violates the upper bound and *UBvalid* is true, that new solution is automatically disregarded.

The costs for the solutions to all cases are presented in Table 2. It can be noted that the solution for Case 1 is the cheaper if one considers only the costs associated to the HEN. However, that solution is the most expensive if Eco-costs are considered as well. The opposite occurs in Case 2 solution, where HENC were increased by 79.9% for yielding a 55.4% reduction in EC. That can be considered a poor economic performance regarding HENC. As expected, the solution with the lowest sum of HENC and EC is the one where those two values were considered in the objective function (Case 3). Although the EC are greater than in Case 2, that value is considerably lower than in Case 1 (51.5%), while the HENC are 16.6% greater, which are more appealing values than those for Case 2. In Case 4, it is worth observing that the metaheuristic method was able to identify a solution at the upper bound proposed. The hypothetical environmental policy is thus respected at a rather small HENC increase in comparison to Case 1 (4.4%).

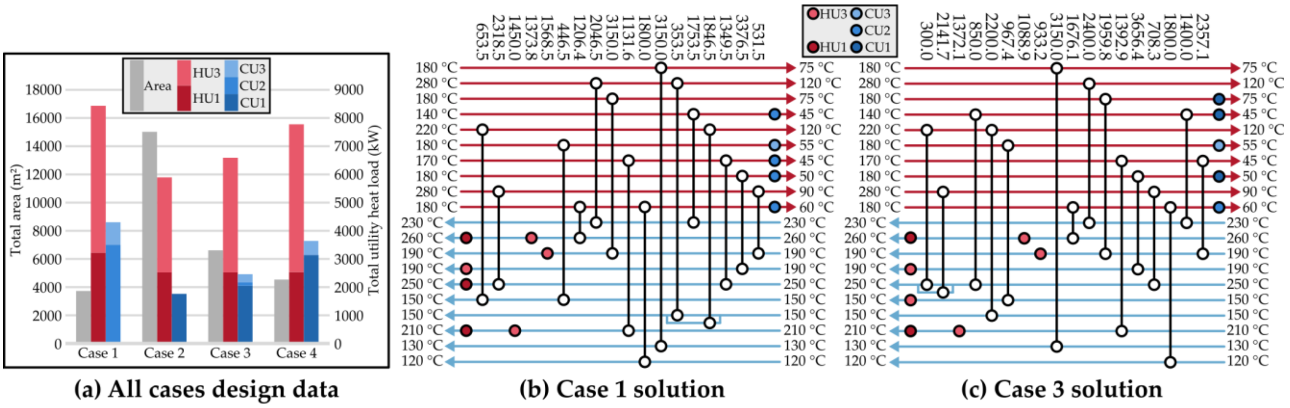
Table 2. Costs and eco-costs for all cases' solutions

Solution	HENC+EC ($/y)	HENC ($/y)	Comparison to Case 1 (HENC)	EC ($/y)	Comparison to Case 1 (EC)
Case 1	10,843,881	2,113,345	0.0%	8,730,536	0.0%
Case 2	7,694,618	3,802,512	+79.9%	3,892,106	-55.4%
Case 3	6,699,658	2,464,501	+16.6%	4,235,157	-51.5%
Case 4	6,706,815	2,206,815	+4.4%	4,500,000	-48.5%

Figure 3 (a) presents design data for all solutions. It is worth noting that even though it has greater utility requirements, area requirements are smaller for Case 1, which presented the lowest HENC. Such greater utility use, however, favors the increase in EC. In Case 2, it can be observed that the area requirement is the greater among all cases' solutions. Such greater area yields a large HEN investment, but also greatly reduces the use of utilities, therefore improving the environmental performance of that solution. The Eco-cost method was followed when solving Case 3 and, thus, the solution found in Case 3 is the optimal choice regarding economic and environmental performance. The optimal solution constraining eco-costs to 4.5M$/y (Case 4) led to a solution that is a middle-ground between those for Cases 1 and 3.

Regarding the types of utilities, it can be noted that HU1 (high pressure steam from coal-fired boiler) is only slightly reduced in more eco-friendly solutions, despite being the utility that causes more environmental impacts. Such choice is thermodynamically-driven. That is, given its high temperature, HU1 is the only utility that can be used to heat some of the cold streams. Hence, it is mandatory that it is present at least for heating such streams. Concerning cold utilities, the use of cooling towers is favored in solutions with lower EC. Regarding area-related Eco-costs, it is worth noting that those are small when compared to utilities-related, and thus, even the high-area solution of Case 2 has a smaller value for Eco-cost, as seen in Table 2.

Figure 3 (b) and (c) present the configurations that can be considered the two more meaningful for the methodology used here. Those are the ones found in Case 1 and Case 3, which have objective functions of HENC and HENC+EC, *i.e.* with and without activating the Eco-costs method. The configurations differ slightly. Note that in Case 3 solution, there is one more heater using HU3 (which is produced from natural gas) than in Case 1 solution. Moreover, Case 3 configuration favors the use of cooling water from cooling towers, which yield less Eco-cost.



(a) All cases design data

(b) Case 1 solution

(c) Case 3 solution

Figure 3. Design data chart for all cases (a) and solutions for Case 1 (b) and Case 3 (c). Numbers above grid diagrams are heat loads, in kW

5 Conclusions

This work presented a framework for the evaluation of heat exchanger networks regarding their economic and environmental performances. Such task was accomplished with aid of the Eco-costs approach, which is based on the Life Cycle Assessment concept and is used to translate environmental impacts into monetary values. That enables the development of single objective optimization models for simultaneously reducing HEN costs and Eco-costs. The methodology was applied to a large-scale example from the literature, evaluated in four different cases. In the solutions, it was observed that the method was able to reduce the coal-fired boiler use (which yields more eco-costs) up to the point its choice was thermodynamically necessary, since it was the only source of high temperature steam. As expected, since cooling towers EC value per kilowatt is lower, it was observed that solutions with lower EC favored their use instead of chillers or compressors. It was noted that heat exchanger construction eco-costs had little influence on the final environmental performance, although heat exchanger areas account greatly for HEN costs.

The metaheuristic-based framework here presented was efficient in finding near-optimal solutions to the model and cases proposed, including when an additional upper-bound was added. That indicates that the use of metaheuristics for HEN can be further explored and that these approaches are an interesting alternative for HEN design considering aspects other than its economic evaluation. As demonstrated, such framework can be efficiently used for proposing networks that are more environmentally friendly. Moreover, it can be employed when a project is required to meet a pre-determined environmental performance due to environmental policies.

6 Acknowledgements

The authors gratefully acknowledge the financial support from Coordination for the Improvement of Higher Education Personnel (CAPES, Brazil) and the National Council for Scientific and Technological Development (CNPq, Brazil).

References

[1] T.F. Yee, I.E. Grossmann, Simultaneous optimization models for heat integration—II. Heat exchanger network synthesis, *Comput. Chem. Eng.* 14 (1990) 1165–1184. doi:10.1016/0098-1354(90)85010-8.

[2] F. Peng, G. Cui, Efficient simultaneous synthesis for heat exchanger network with simulated annealing algorithm, *Appl. Therm. Eng.* 78 (2015) 136–149. doi:10.1016/j.applthermaleng.2014.12.031.

[3] Z. Huo, L. Zhao, H. Yin, J. Ye, Simultaneous synthesis of structural-constrained heat exchanger networks with and without stream splits, *Can. J. Chem. Eng.* 91 (2013) 830–842. doi:10.1002/cjce.21702.

[4] M.A.S.S. Ravagnani, A.P. Silva, P.A. Arroyo, A.A. Constantino, Heat exchanger network synthesis and optimisation using genetic algorithm, *Appl. Therm. Eng.* 25 (2005) 1003–1017. doi:10.1016/j.applthermaleng.2004.06.024.

[5] A.P. Silva, M.A.S.S. Ravagnani, E.C. Biscaia, J.A. Caballero, Optimal heat exchanger network synthesis using particle swarm optimization, *Optim. Eng.* 11 (2010) 459–470. doi:10.1007/s11081-009-9089-z.

[6] L.V. Pavão, C.B.B. Costa, M.A.S.S. Ravagnani, Automated heat exchanger network synthesis by using hybrid natural algorithms and parallel processing, *Comput. Chem. Eng.* 94 (2016) 370–386. doi:10.1016/j.compchemeng.2016.08.009.

[7] L.V. Pavão, C.B.B. Costa, M.A.S.S. Ravagnani, Heat Exchanger Network Synthesis without stream splits using parallelized and simplified simulated Annealing and Particle Swarm Optimization, *Chem. Eng. Sci.* 158 (2017) 96–107. doi:10.1016/j.ces.2016.09.030.

[8] L.V. Pavão, C.B.B. Costa, M.A.S.S. Ravagnani, L. Jiménez, Large-scale heat exchanger networks synthesis using simulated annealing and the novel rocket fireworks optimization, *AIChE J.* 63 (2017) 1582–1601. doi:10.1002/aic.15524.

[9] Z. Bao, G. Cui, J. Chen, T. Sun, Y. Xiao, A novel random walk algorithm with compulsive evolution combined with an optimum-protection strategy for heat exchanger network synthesis, *Energy*. (2018). doi:10.1016/j.energy.2018.03.170.

[10] H. Zhang, G. Cui, Optimal heat exchanger network synthesis based on improved cuckoo search via Lévy flights, *Chem. Eng. Res. Des.* 134 (2018) 62–79. doi:10.1016/J.CHERD.2018.03.046.

[11] L.A. López-Maldonado, J.M. Ponce-Ortega, J.G. Segovia-Hernández, Multiobjective synthesis of heat exchanger networks minimizing the total annual cost and the environmental impact, *Appl. Therm. Eng.* 31 (2011) 1099–1113. doi:10.1016/j.applthermaleng.2010.12.005.

[12] P. Vaskan, G. Guillén-Gosálbez, L. Jiménez, Multi-objective design of heat-exchanger networks considering several life cycle impacts using a rigorous MILP-based dimensionality reduction technique, *Appl. Energy.* 98 (2012) 149–161. doi:10.1016/j.apenergy.2012.03.018.

[13] M.A.S.S. Ravagnani, T.B. Mano, E.P. Carvalho, A.P. Silva, C.B.B. Costa, Multi-objective Heat Exchanger Networks Synthesis Considering Economic and Environmental Optimization, *Comput. Aided Chem. Eng.* 33 (2014) 1579–1584. doi:10.1016/B978-0-444-63455-9.50098-2.

[14] L.V. Pavão, C.B.B. Costa, M.A.S.S. Ravagnani, L. Jiménez, Costs and environmental impacts multi-objective heat exchanger networks synthesis using a meta-heuristic approach, *Appl. Energy.* 203 (2017) 304–320. doi:10.1016/j.apenergy.2017.06.015.

[15] T.B. Mano, L. Jiménez, M.A.S.S. Ravagnani, Incorporating life cycle assessment eco-costs in the optimization of heat exchanger networks, *J. Clean. Prod.* 162 (2017) 1502–1517. doi:10.1016/j.jclepro.2017.06.154.

[16] The Model of the Eco-costs / Value Ratio (EVR). www.ecocostsvalue.com (accessed December 1, 2017).

[17] K.C. Furman, N. V. Sahinidis, Computational complexity of heat exchanger network synthesis, *Comput. Chem. Eng.* 25 (2001) 1371–1390. doi:10.1016/S0098-1354(01)00681-0.

[18] X. Luo, Q.-Y. Wen, G. Fieg, A hybrid genetic algorithm for synthesis of heat exchanger networks, *Comput. Chem. Eng.* 33 (2009) 1169–1181. doi:10.1016/j.compchemeng.2008.12.003.

Energy efficient scheduling of a multi-states and multi-speeds single machine system

MohammadMohsen Aghelinejad, Yassine Ouazene, Alice Yalaoui

Industrial Systems Optimization Laboratory (ICD-CNRS 6281)
Université de Technologie de Troyes, France
(mohsen.aghelinejad, yassine.ouazene, alice.yalaoui)@utt.fr

Electricity, as one of the main used energy sources in production systems, plays a very important role in modern industries. The rapid and ongoing growth of electricity price has become an important issues for many countries. Therefore, improving the energy efficiency of a production system, saving the costs involved and its environmental impact, has encouraged many researchers all around the world to study these issues. For example in [4], the authors studied the scheduling problem of processing jobs with arbitrary power demands on an uniform speed or speed-scalable single machine to minimize total electricity cost. The production scheduling of a single machine with different states is studied in [5] to minimize total energy consumption costs when the price of energy varies between the periods. Two versions of this problem with/without fixed sequence of the jobs are addressed in [1] and [2]. Moreover, the complexity of the preemptive version of the problem without fixed sequence of the jobs is analysed in [3] based on a dynamic programming approach.

In this paper, the energy efficient scheduling of n jobs during T periods on a speed-scalable single machine is addressed. The machine has 3 main states (ON,OFF, and Idle), and two transition states (named Ton and Toff)(Figure 1). In each state $s \in \{ON,\ OFF,\ Idle,\ Ton,\ Toff\}$, the machine consumes a specific amount of energy (e_s) per period. The energy consumption in state OFF is equal to 0 ($e_{OFF} = 0$), and the machine must be in OFF states during the initial and final periods. The energy consumption of the machine during state ON depends on the processed job. For each state s, the machine requires a specific number of periods d_s; $s \in \{ON,\ OFF,\ Idle,\ Ton,\ Toff\}$. In the speed-scalable case, the machine has different possibilities for processing each job with different speeds. So, each job $j = 1, ..., n$ has v_j possible processing times defined in a set as $P_j = \{p_{j,1}, \cdots, p_{j,v_j}\}$. Their corresponding energy consumptions $q_{j,i}$ to each $p_{j,i}$ ($j = 1, ..., n,\quad i = 1, ..., v_j$) are defined in sets $Q_j = \{q_{j,1}, \cdots, q_{j,v_j}\}$ for $j = 1, ..., n$. Since performing a job more faster needs more energy consumption, the following relations are assumed:

$$p_{j,1} > p_{j,2} > \cdots > p_{j,v_j} \quad ; \quad q_{j,1} < q_{j,2} < \cdots < q_{j,v_j} \quad ; \forall j \in \{1, \cdots, n\} \tag{1}$$

The objective of this study is to find the most economical production schedule in terms of total energy consumption costs when Time-Of-Use electricity tariff is considered for each period. To the best of our knowledge, our study is the first work in the literature that addresses the energy efficiency of a multi-states and multi-speeds single machine system with the time-dependent electricity cost.

In [4], the authors proved that the non-preemptive scheduling problem of a speed-scalable single machine is NP-hard. So, the non-preemptive scheduling problem of a multi-states and speed-scalable single machine, which is presented in this paper, is also NP-hard. Therefore, a genetic algorithm and a memetic algorithm are proposed to solve the problem in a reasonable time (especially for the large size instances). Any solution of this problem consists in T periods, from 0 to T, for which the machines state or the processed job is specified. So, each chromosome is represented by $T + 1$ genes and each gene identifies the machines state in a period. To distinguish between the machine's states, each state is represented by a specific number as $OFF = 1$, $Ton = 2$, $Idle = 3$, and $Toff = 4$. Besides, an integer number greater than 100 ($k > 100$) indicates that the machine is in ON state. If in period t, the machine processes job j with speed i, in its corresponding chromosome, the gene t fills with number $(100 * j + 10 * i)$. Figure 3 represents the corresponding chromosome of the instance of Figure 2. Since in this instance the number of periods is 32, so this chromosome consists of 33 genes. The number 230 in 10th gene means that during period 9 the machine processes job 2 with speed 3. Also, the number 4 in 27th gene means that during period 26 the machine is in Toff state.

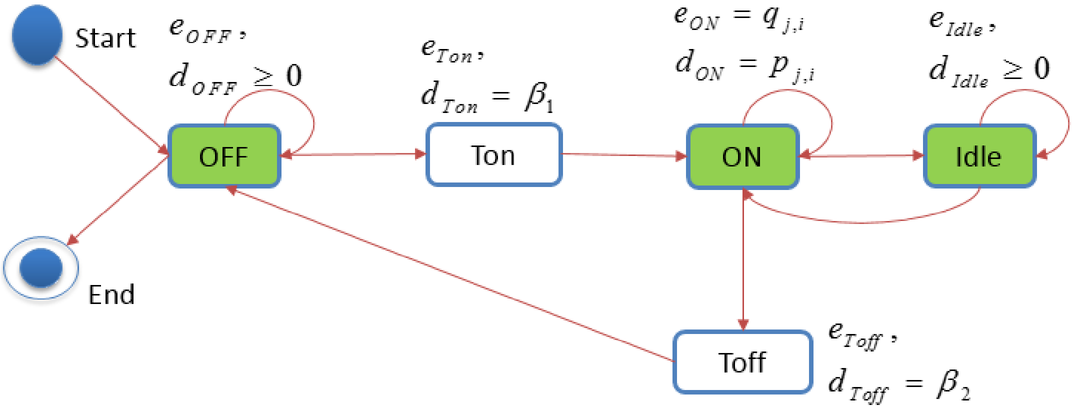
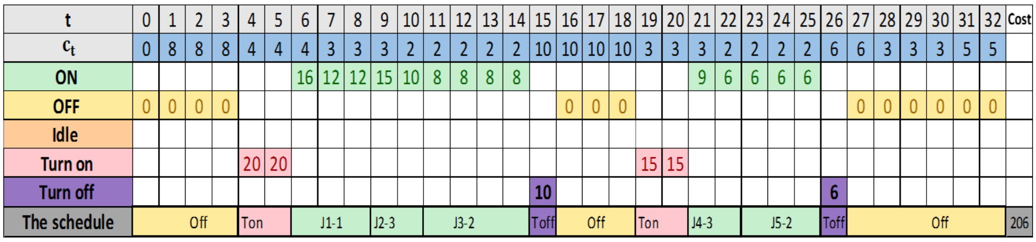
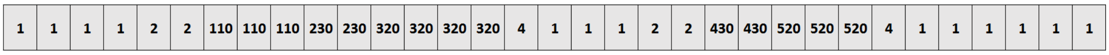


Fig. 1. Machine states and possible transitions.



t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	Cost
c_t	0	8	8	8	4	4	4	3	3	3	2	2	2	2	2	10	10	10	10	3	3	3	2	2	2	2	6	6	3	3	3	5	5	
ON							16	12	12	15	10	8	8	8	8							9	6	6	6	6								
OFF	0	0	0	0													0	0	0									0	0	0	0	0	0	
Idle																																		
Turn on					20	20														15	15													
Turn off																10											6							
The schedule	Off				Ton		J1-1			J2-3		J3-2				Toff	Off			Ton		J4-3		J5-2			Toff	Off						206

Fig. 2. Illustrative example

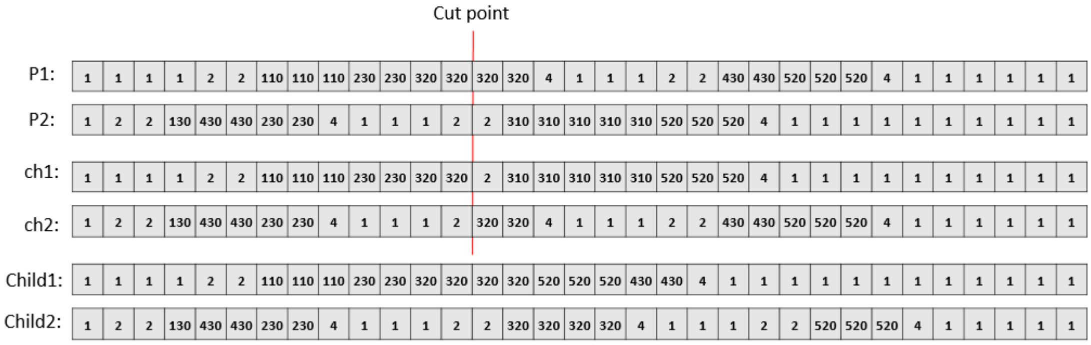


1	1	1	1	2	2	110	110	110	230	230	320	320	320	320	4	1	1	1	2	2	430	430	520	520	520	4	1	1	1	1	1	1

Fig. 3. The genetic algorithm's chromosome encoding

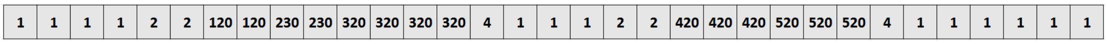
In this study, the roulette wheel selection has been chosen as parents selection operator, and the main parameters of genetic algorithm such as: population size, crossover rate, mutation rate, crossover operator, mutation operator and number of iterations, are determined by using the Taguchi method. As a result, the initial population size and the number of iterations are considered equal to 150 and 100, respectively. The single-point is selected as crossover method, and the crossover rate is selected equal to 0.7 (70%). The cut point will be randomly generated from the periods index. Then, the first children will be composed of the first parent from the beginning to the cut point, and the second parent from the first gene after the cut point to the end of the chromosome, and the reverse for the second children. After producing the children, a correction procedure must be done to convert the not feasible solutions to the feasible one (see Figure 4). Besides, for the mutation method, a chromosome from the initial population will be randomly selected and the mutation will be performed on the selected gene by swapping its value. Then, a correction procedure must be done on the obtained offsprings. The mutation rate is selected equal to 0.1 (10%). Finally, the best chromosomes in terms of the fitness function must be selected from all the initial population and the obtained chromosomes by crossover and mutation methods, to update the initial population for the next iteration.

From this genetic algorithm, a local search procedure is applied to get a memetic algorithm. The local search may increase the quality of the 15 (10% of the population size) best chromosomes of the population in each iteration. For each of them, the neighborhood is explored by increasing one unit the processing speed of one job (if it's processing speed is less than the maximum speed), and consequently, processing all the remaining jobs earlier. This procedure must be repeated for all the jobs in their sequence order. So, at most n new solutions can be created from each initial



P1:	1	1	1	1	2	2	110	110	110	230	230	320	320	320	320	4	1	1	1	2	2	430	430	520	520	520	4	1	1	1	1	1	1
P2:	1	2	2	130	430	430	230	230	4	1	1	1	2	2	310	310	310	310	310	520	520	520	4	1	1	1	1	1	1	1	1	1	1
ch1:	1	1	1	1	2	2	110	110	110	230	230	320	320	2	310	310	310	310	310	520	520	520	4	1	1	1	1	1	1	1	1	1	1
ch2:	1	2	2	130	430	430	230	230	4	1	1	1	2	320	320	4	1	1	1	2	2	430	430	520	520	520	4	1	1	1	1	1	1
Child1:	1	1	1	1	2	2	110	110	110	230	230	320	320	320	320	520	520	520	430	430	4	1	1	1	1	1	1	1	1	1	1	1	1
Child2:	1	2	2	130	430	430	230	230	4	1	1	1	2	2	320	320	320	320	4	1	1	1	2	2	520	520	520	4	1	1	1	1	1

Fig. 4. single-point crossover method



1	1	1	1	2	2	120	120	230	230	320	320	320	320	4	1	1	1	2	2	420	420	420	520	520	520	4	1	1	1	1	1	1

Fig. 5. The memetic algorithm's chromosome encoding

solution, and finally, the solution which has the best objective value in the neighborhood replace to the initial one. For example, for the considered solution in Figure 3, the first job is processed with the first speed (3 periods). So, in the first solution proposed by memetic algorithm, the first job is processed with the second speed (2 periods), and all the remaining jobs are performed one period earlier during the rest of periods.

The performance of the proposed methods have been tested based on several numerical experiments. Five different examples are randomly generated for each instance size by changing the processing times and the energy consumptions of the jobs, as well as the unit of energy price in each period. The genetic algorithm (GA) and the memetic algorithm (MA) have been coded by C++ language in the Visual Studio 2015, and CPLEX software is used to solve the instances with the exact method (Branch and Cut). The computation time for CPLEX was set to 3600 seconds. As a result, for the small size problems (the instances with less than 15 jobs, 5 speeds, and 120 periods), the genetic algorithm and the memetic algorithm find the solutions with the average gap of 7.5% and 2.7% to the optimal solution with the computation time of 15.64 s and 18.50 s. For the large size instances, the memetic algorithm improves the results of the genetic algorithm in average 18%, and the average computation times are equals to 34.37 s for the GA and 61.59 s for the MA.
For the future works, we will attempt to propose a more efficient mathematical model for this problem. We are also interested to investigate the performance of the other exact methods like Branch and Bound, and Bender's decomposition for this problem.

References

1. Aghelinejad, M., Ouazene, Y., and Yalaoui, A. : Machine and production scheduling under electricity time varying prices. In Industrial Engineering and Engineering Management (IEEM), 2016 IEEE International Conference on, 992-996. IEEE.
2. Aghelinejad, M., Ouazene, Y., and Yalaoui, A. : Production scheduling optimization with machine state and time dependent energy costs. International Journal of Production Research, 2017, 1-18.
3. Aghelinejad, M., Ouazene, Y., and Yalaoui, A. : Preemptive scheduling of a single machine with finite states to minimize energy costs. International conference on Optimization and Decision Science (ODS), 2017
4. Fang, K., Uhan, N.A., Zhao, F., and Sutherland, J.W. : Scheduling on a single machine under time-of-use electricity tariffs. Annals of Operations Research, 1-29.
5. Shrouf, F., Ordieres-Meré, J., García-Sánchez, A., and Ortega-Mier, M. : Optimizing the production scheduling of a single machine to minimize total energy consumption costs. Journal of Cleaner Production, 67, 197-207.

Meta-heuristics for global reliability optimization of solder joints in electronic devices

H. Hamdani[1,2], A. El Hami[1] and B. Radi[2]

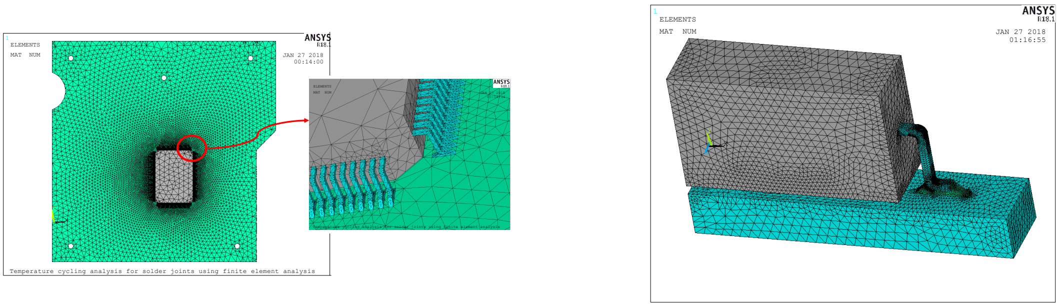
[1] Laboratoire de Mécanique de Normandie (LMN), INSA Rouen, France
hamid.hamdani@insa-rouen.fr
abdelkhalak.elhami@insa-rouen.fr
[2] Laboratoire d'Ingénieuré Management Industriel et Innovation (IMII),FST Settat Maroc
bouchaib.radi@yahoo.fr

Abstract

Electronics are increasingly used in controlled and embedded mechanical systems. This leads to new mechatronics devices which are lighter, smaller and use less energy. However, this mechatronics approach must ensure a smooth operation. The reliability assessment of these systems remains a major challenge. However, their failure are often caused by many extreme solicitations of thermal(temperature variation), and mechanical nature (shocks, vibration), etc. The most failures of mechatronic devices are caused by solder joints fatigue. consequently, to meet the reliability requirements, the robustness of the solder joints should be validated and reliably optimized. In fact, due to thermal expansion, thermal cycles induce mechanical stresses in the solder joints connecting the electronic components to the PCB. Therefore, these stresses can be significant leading to plastic deformation whose accumulation can cause damage leading to solder joint failure and consequently, the failure of the complete device. In order to maximize the number of fatigue life cycles and to avoid the failures in the operational environment, the robust optimization and reliability based design optimization of critical solder joints should be carried out in the design process of mechatronic devices.

In the RBDO [1, 2], the derivative-based algorithms are the most used methods [3], which require derivative computation. The main advantage of those methods is that they need a much smaller number of iterations to converge to an optimum over other methods. However, only convergence towards a local minimum is guaranteed. The derivative-free algorithms which are based only on original fitness function [4], are proved as powerful tools for global optimum research and therefore are widely used in real word problem such as engineering optimization. The evolution strategies(ES) are one among powerful derivative free algorithms, which are a popular methods for black-box optimization, where no expressions of so-called objective functions, are known and no derivatives can be computed [5]. The use of Evolution strategies in real word problems proved their power [6].

In the first stage of this work, the virtual thermo-mechanical test [7] was performed to evaluate the reliability of solder joints of a mechatronic device[8]. The 3D FE developed model(figure1) takes into account the nonlinearities properties of viscoplastique behavior of the solder joints[9]. This study reveals the interest to use the meta modeling techniques with CMA-ES, to increase the reliability of solder joints of the mechatronic devices



(a)

(b)

Figure 1: Global (a) and local model (b) of the solder joints

.

The second stage aims to applied the CMA-ES algorithm [10] to the global optimization of solder joints in the mechatronic system (figure 1), in order to optimize the thermo-mechanical performance of solder joints by maximizing the number of life cycles.The objective function (number of life cycles based on design parameters) is evaluated repeatedly by performing iterations of CMA-ES algorithm.The interest of this method resides in its efficiency to calculates a global optimum of the objective function. The results obtained show an increase in the number of life cycles of the solder joints, and signifies a success and the robustness of the CMAES method in the mechatronics field.

The last stage of this work intended to constraint the CMA-ES Algorithm with global reliability methods[11] in order to develop a new Robust Reliability based design optimization, those methods are based on global approximation model of performance function using Kriging Metamodel. Metamodeling technique[12] is the basis for these global reliability methods. This category of methods, firstly, approximates the performance function by Kriging metamodel, and then perform sampling methods based on the built surrogate model to calculate the failure probability. The computational cost is significantly reduced with the aid of metamodel, since the metamodel is cheap to evaluate. Additionally, global reliability methods can give accurate results because Kriging metamodel can adequately model the nonlinear limit state function.

References

1. El Hami, A. and Radi, B. (2011). Comparison study of different reliability-based design optimization approaches. In Advanced Materials Research, volume 274, pages 113–121. Trans Tech Publ.
2. El Hami, A. and Radi, B. (2013). Uncertainty and optimization in structural mechanics.John Wiley and Sons.
3. El Maani, R., Makhloufi, A., Radi, B., and El Hami, A. (2018). Reliability-based design optimization with frequency constraints using a new safest point approach. Engineering Optimization, pages 1–18
4. Talbi, E.-G. (2009). Metaheuristics: from design to implementation, volume 74.John Wiley and Sons.
5. Kramer, O. (2014). A Brief Introduction to Continuous Evolutionary Optimization. Springer.
6. Bäck, T., Foussette, C., and Krause, P. (2013). Contemporary evolution strategies. Springer.
7. Aoues, Y., Makhloufi, A., Pougnet, P., and El Hami, A. (2012). Probabilistic assessment of thermal fatigue of solder joints in mechatronic packaging. In Proceedings of the 1st International Symposium on Uncertainty Quantification and Stochastic Modeling, Maresias, SP, Brazil.
8. El Hami, A. and Pougnet, P. (2015). Embedded Mechatronic Systems, Volume 2. Elsevier.
9. Anand, L. (1982). Constitutive equations for the rate-dependent deformation of metals at elevated temperatures. Journal of Engineering Materials and Technology(Transactions of the ASME), 104(1):12–17.
10. Hansen, N. (2016). The cma evolution strategy: A tutorial. arXiv preprint arXiv:1604.00772.
11. Echard B., Gayton N., and Lemaire M., AK-MCS: an active learning reliability method combining Kriging and Monte Carlo simulation. Structural Safety, vol. 33, no. 2, pp. 145-154, 2011.
12. Hamdani, H., Radi, B., and El Hami, A. (2017). Métamodélisation pour une conception robuste des systèmes mécatroniques. Incertitudes et fiabilité des systèmes multiphysiques, 2(Numéro 2):10.

Dynamic Programming heuristic for k-means Clustering among a 2-dimensional Pareto Frontier

N. Dupin[1], F. Nielsen[2,3], and E. Talbi[1]

[1] Univ. Lille, UMR 9189 - CRIStAL - Centre de Recherche en Informatique Signal et Automatique de Lille, F-59000 Lille, France
nicolas.dupin@polytechnique.edu,el-ghazali.talbi@univ-lille1.fr
[2] On leave of absence from École Polytechnique, LIX, Palaiseau, France
[3] Sony Computer Science Laboratories Inc, Tokyo, Japan
Frank.Nielsen@acm.org

Abstract. This paper is motivated by a real life application of multi-objective optimization without preference. Having many incomparable solutions with Pareto optimality, the motivation is to select a small number of representative solutions for decision makers. The k-means clustering is investigated in this paper for the 2-dimensional case using the specific property that the points to cluster are Pareto optimal in $\mathbb{R}^2$. A dynamic programming heuristic is designed, assuming a specific property of the optimal clusters. Having N points to partition in K cluster, the complexity of the heuristic is in $O(N^3)$. If the presented conjecture is proven, it induces that K-means clustering would be polynomially solvable in $O(N^3)$. The dynamic programming algorithm can be adapted to consider cardinality constraints for the clusters, which can improve the complexity. Matheuristics can be derived from the previous algorithm to have a complexity in $O(K.N^2)$. These algorithms allows a natural parallel implementation. A posteriori, the complexity of the clustering algorithms allows also to consider these algorithms inside multi-objective meta-heuristics to archive diversified non-dominated points along the Pareto front. Applications in stochastic/robust optimization are also discussed, clustering scenarios of uncertainty.

Keywords: Clustering algorithms; bi-objective optimization ; Pareto frontier; k-means ; dynamic programming; matheuristics ; data science

1 Introduction

This paper is motivated by real-life applications of multi-objective optimization [7, 21]. The solutions of multi-objective optimization approaches are a set of efficient solutions, non dominated considering the Pareto optimality. It is a weak preference rule, many efficient solutions may exist. The problematic is here to select for decision makers only a few good compromise solutions K from $N \gg K$ non dominated solutions. This selection aims to maximize the representativity of these K solutions among the N initial ones. This can be seen as an application of clustering algorithms [14], partitioning the N elements into K subsets with a maximal similarity, and giving a representative (or central) element of the optimal clusters.

K-means clustering is one of the most famous unsupervised learning problem, and is widely studied in the literature. A seminal algorithm to solve K-means problems was provided by Lloyd in [16], a steepest descent heuristic that allows easily parallel computations with modern computers [18]. Hartigan's heuristic improve the quality of local minimums [12, 24]. We note that recent works investigates larger neighborhoods [19]. A careful initialization has also an important impact to solve k-means [2]. K-means was proven to be NP hard [5]. Special cases of K-means are also been proven NP-hard in a general Euclidean space: the problem is still NP-hard when the number of clusters is 2 ([1]), or when the dimensionality is 2 ([17]). The case $K = 1$ is trivially polynomial. The 1-dimensional case was proven polynomially solvable thanks to a dynamic programming algorithm in [27], with a complexity in $O(k.n^2)$ with a dynamic programming algorithm. This last algorithm was improved in [11], for a dynamic programming algorithm with a complexity in $O(kn)$ using memory space in $O(n)$. Being in a general Euclidean space, K-means can be solved by a Polynomial Time approximation Schemes (PTAS), i.e. algorithms allowing to have a $1 + \varepsilon$ approximation solvable in polynomial time for all $\varepsilon > 0$, as developed in [3].

This paper develops dynamic programming heuristics to solve the particular case of k-means clustering in a 2-dimensional Pareto frontier. A specific propriety of the optimal clusters is assumed in a conjecture. If the conjecture is proven, this paper proves that the problem would be polynomially solvable in $O(N^3)$ to optimality. At this stage, it is only a heuristic. Matheuristics can also be derived from the previous algorithm in $O(K.N^2)$.

2 Problem statement and notations

We suppose in this paper having a set $E = \{x_1, \dots, x_N\}$ of N elements of $\mathbb{R}^2$, such that for all $i \neq j$, $x_i \mathcal{I} x_j$ defining the binary relations $\mathcal{I}, \prec$ for all $y = (y^1, y^2), z = (z^1, z^2) \in \mathbb{R}^2$ with:

$$y \mathcal{I} z \Longleftrightarrow y \prec z \text{ or } z \prec y \tag{1}$$

$$y \prec z \Longleftrightarrow y^1 < z^1 \text{ and } y^2 > z^2 \tag{2}$$

This property is verified in the applicative context, E being the solution of a bi-objective optimization problem without preference. This applies for exact approaches or population meta-heuristics like evolutionary algorithms and others [26].

We consider in this paper the Euclidian distance, defining for all $y = (y^1, y^2), z = (z^1, z^2) \in \mathbb{R}^2$:

$$d(y,z) = \|y - z\| = \sqrt{(y^1 - z^1)^2 + (y^2 - z^2)^2} \tag{3}$$

We define $\mathit{\Pi}_K(E)$, as the set of all the possible partitions of E in K subsets:

$$\mathit{\Pi}_K(E) = \left\{ P \subset \mathcal{P}(E) \,\middle|\, \forall p, p' \in P,\ p \cap p' = \emptyset \text{ and } \bigcup_{p \in P} = E \text{ and } \mathrm{card}(P) = K \right\} \tag{4}$$

K-means clustering is a combinatorial optimization problems indexed by $\mathit{\Pi}_K(E)$. K-means clustering minimizes the sum for all the K clusters of the average distances from the points of the clusters to the centroid. Mathematically, this can be written as:

$$\min_{\pi \in \mathit{\Pi}_K(E)} \sum_{p \in \pi} \frac{1}{\mathrm{card}(p)} \sum_{x \in p} \left\| x - \frac{1}{\mathrm{card}(p)} \sum_{y \in p} y \right\|^2 \tag{5}$$

3 Preliminary results

This section gives some preliminary results necessary for the following developments.

Proposition 1 (Order Properties) $\prec$ *has following properties:*

$$\forall x, y, z \in \mathbb{R}^2,\ x \prec y \text{ and } y \prec z \Longrightarrow x \prec z \tag{6}$$

$$\forall x, y, z \in \mathbb{R}^2, \lambda \in [0,1]\ x \prec z \text{ and } y \prec z \Longrightarrow \lambda x + (1-\lambda) y \prec z \tag{7}$$

$$\forall x, y, z \in \mathbb{R}^2, \lambda \in [0,1]\ x \prec y \text{ and } x \prec z \Longrightarrow x \prec \lambda z + (1-\lambda) y \tag{8}$$

Proof: The proofs of these results are easy, straightforward applications of the properties of the relation $<$ in $\mathbb{R}$.

Proposition 2 (Total order) *Points (x_i) can be indexed such that:*

$$\forall (i_1, i_2) \in [\![1; N]\!]^2,\ i_1 < i_2 \Longrightarrow x_{i_1} \prec x_{i_2} \tag{9}$$

Proof: We prove the result by induction on $N \in \mathbb{N}$.
For $N = 1$, the propriety (9) is trivially verified.
Let us suppose $N > 1$ and the Induction Hypothesis (IH) that (9) is true for $N - 1$.
Let $A = \{a \in \mathbb{R} \mid \exists x_i \in E,\ a = x_i^1\}$.
A is a finite subset of $\mathbb{R}$, it has a maximum. Let m such that $x_m^1 = \max A$.

Let $m' \neq m$. $x_m \,\mathcal{I}\, x_{m'}$ with the definition of E, it implies $x^1_{m'} \prec x^1_m$ or $x^1_m \prec x^1_{m'}$.
$x^1_m \prec x^1_{m'}$ implies $x^1_{m'} > x^1_m$ which is in contradiction with $x^1_m = \max A$, thus $x^1_{m'} \prec x^1_m$.
It proves:

$$\forall i \in [\![1;N]\!] - \{m\},\ \ x_i \prec x_m \tag{10}$$

Applying (IH) to $[\![1;N]\!] - \{m\}$ allows to index $[\![1;N]\!] - \{m\}$ as $i_1 < \cdots < i_{N-1}$ with propriety (9).
Defining $i_N = m$, the missing inequalities are furnished by (10) to have the result true for N.
It proves by induction that (9) is true for all $N \in \mathbb{N}$. □

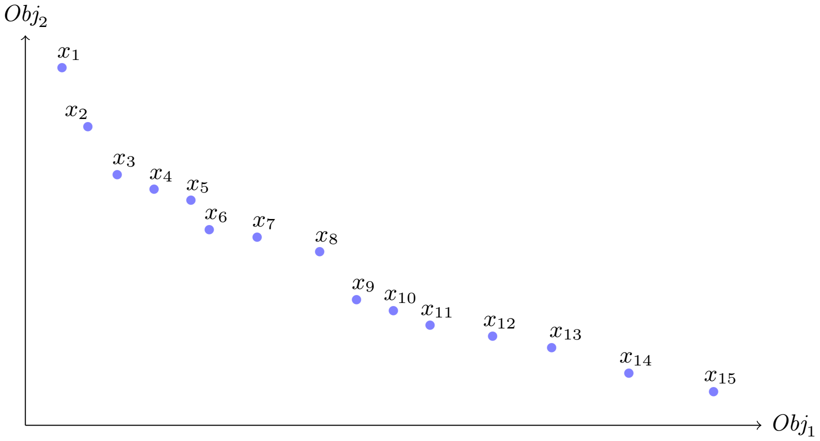


Fig. 1. Illustration of the indexation implied by Proposition 2 in a 2-d Pareto front

Proposition 3 *We suppose that points (x_i) are sorted following Proposition 2. Let $(i_1, i_2, i_3) \in [\![1;n]\!]^3$.*

$$i_1 < i_2 < i_3 \Longrightarrow d(x_{i_1}, x_{i_2}) < d(x_{i_1}, x_{i_3}) \tag{11}$$

Proof: Let $i_1 < i_2 < i_3$. Proposition 2 ordering ensures $x^1_{i_1} < x^1_{i_2} < x^1_{i_3}$ and $x^2_{i_1} > x^2_{i_2} > x^2_{i_3}$)
$d(x_{i_1}, x_{i_2})^2 = (x^1_{i_1} - x^1_{i_2})^2 + (x^2_{i_1} - x^2_{i_2})^2$
With $x^1_{i_3} - x^1_{i_1} > x^1_{i_2} - x^1_{i_1} > 0$, $(x^1_{i_1} - x^1_{i_2})^2 < (x^1_{i_1} - x^1_{i_3})^2$
With $x^2_{i_3} - x^2_{i_1} < x^2_{i_2} - x^2_{i_1} < 0$, $(x^2_{i_1} - x^2_{i_2})^2 < (x^2_{i_1} - x^2_{i_3})^2$
Thus $d(x_{i_1}, x_{i_2})^2 < (x^1_{i_1} - x^1_{i_3})^2 + (x^2_{i_1} - x^2_{i_3})^2 = d(x_{i_1}, x_{i_3})^2$. □

4 Optimal clusters for *k*-means clustering in a 2d-Pareto Front

This section gives the fundamental results for the algorithm of section 5. The optimal clusters can be characterized and enumerated polynomially, as proven in this section.

4.1 Conjecture for the optimality of clusters

This result is intuitive geometrically. However, it is not fully proven, so we mention it as a conjecture.

Conjecture 1 (Optimal clusters) *We suppose that points (x_i) are sorted following Proposition 2. We conjecture we have optimal solutions of the minimization problem 5 having only clusters $\mathcal{C}_{i,i'} = \{x_j\}_{j \in [\![i,i']\!]} = \{x \in E \mid \exists j \in [\![i,i']\!],\ x = x_j\}$*

This characterization of the clusters is crucial for an application of the Dynamic programming algorithm with a Bellman's optimality property.

4.2 Computing cost for one single cluster

This section investigates the complexity to compute the cost of the possible optimal clusters. We note $c_{i,i'}$ the cost of the cluster $\mathcal{C}_{i,i'}$:

$$c_{i,i'} = \frac{1}{i'-i} \sum_{j=i}^{i'} \left\| x_j - \frac{1}{i'-i} \sum_{k=i}^{i'} x_k \right\|^2$$

Computing $c_{i,i'}$ requires using this formula $O(i'-i)$ operations, for the computation of the centroid $\frac{1}{i'-i} \sum_{k=i}^{i'} x_k$ and for the summation over the $i'-i$ elements to compute their distance to the centroid. The other operations are in $O(1)$.

The next section requires to compute all the $c_{i,i'}$ with $i < i'$. Processing all the computations of $c_{i,i'}$ This makes a summed complexity in:

$$\sum_{i<i'} O(i'-i) = O(N^3)$$

Algorithm 1: k-means clustering in a 2d-Pareto Front

```
Input:
- $N$ points of $\mathbb{R}^2$, $E = \{x_1, \dots, x_N\}$ such that for all $i \neq j$, $x_i \ \mathcal{I} \ x_j$ ;
- $K \in \mathbb{N}$ the number of clusters

Cluster2dPareto(E,K)
   //Initialization phase.
   define matrix $c$ with $c_{i,j} = 0$ for all $(i,j) \in [\![1;N]\!]^2$
   initialize matrix $C$ with $C_{i,k} = 0$ for all $i \in [\![0;N]\!], k \in [\![1;K]\!]$
   initialize $\mathcal{P}$ =nil, a set of sub-intervals of $[\![1;N]\!]$.
   sort $E$ following the order of Proposition 2
   compute $c_{i,j}$ for all $(i,j) \in [\![1;N]\!]^2$ as in section 4.2
   //Construction of the matrix $C$
   for $i = 1$ to $N$
      // case $k = 1$ treated separately
      set $C_{i,k} = c_{1,i}$
      for $k = 2$ to $K$
         set $C_{i,k} = \min_{j \in [\![1,i]\!]} C_{j-1,k-1} + c_{j,i}$
      end for
   end for
return $C_{N,K}$ the optimal cost
   //Backtrack phase
   $i = N$
   for $k = K$ to 1 with increment $k \leftarrow k-1$
      find $j \in [\![1,i]\!]$ such that $C_{i,k} = C_{j-1,k-1} + c_{j,i}$
      add $[\![j,i]\!]$ in $\mathcal{P}$
      $i = j-1$
   end for
return the partition $\mathcal{P}$ giving the cost $C_{N,K}$
```

5 Dynamic Programming algorithm

Conjecture 1 and the polynomial computations of all the $c_{i,i'}$ with $i < i'$ allows to derive a dynamic programming algorithm. Defining $C_{i,k}$ as the optimal cost of the k-means clustering with k cluster among points $[\![1,i]\!]$ for all $i \in [\![1,N]\!]$ and $k \in [\![1,K]\!]$, we have following induction relation:

$$\forall i \in [\![1,N]\!], \forall k \in [\![2,K]\!], \quad C_{i,k} = \min_{j \in [\![1,i]\!]} C_{j-1,k-1} + c_{j,i} \tag{12}$$

This last relation use the convention that $C_{0,k} = 0$ for all $k \geqslant 0$. The case $k = 1$ is directly given by:

$$\forall i \in [\![1, N]\!], \quad C_{i,1} = c_{1,i} \tag{13}$$

These relations allow to compute the optimal values of $C_{i,k}$ by dynamic programming in the Algorithm 1. $C_{N,K}$ is the optimal solution of the k-means problem, a backtracking algorithm on the matrix $(C_{i,k})_{i,k}$ allows to compute the optimal partitioning clusters:

Theorem 1. *Let $E = \{x_1, \dots, x_N\}$ a subset of N points of $\mathbb{R}^2$, such that for all $i \neq j$, $x_i \;\mathcal{I}\; x_j$. The complexity of Algorithm 1 is in $O(N^3)$. If the conjecture 1 is proven, the k-means optimal clustering is polynomially solvable with Algorithm 1.*

Proof: In Algorithm 1, it is easy to show by induction that $C_{i,k}$ has its final value for all $i \in [\![1, N]\!]$ at the end of the for loops from $k = 2$ to K. The reason is that the induction formula (12) uses only values $C_{i,j}$ with $j < k$. $C_{N,K}$ is thus at the end of these loops the optimal value of the K-means clustering among the N points of E. The backtracking phase searches for the equalities in $C_{i,k} = C_{j'-1,k-1} + c_{j',i} = \min_{j \in [\![1,i]\!]} C_{j-1,k-1} + c_{j,i}$, proving that such cluster $\mathcal{C}_{j',i}$ allows to give an optimal solution.

Let us analyze the complexity. Sorting and indexing the elements of E following Proposition 2 is equivalent to sort following one dimension, as highlighted in the proof of Proposition 2. The complexity of this sorting phase is thus in $O(N \log N)$. The straightforward computation of the matrix $c_{i,i'}$ has a complexity in $O(N^3)$ following the computation of section 4. The construction of the matrix $C_{i,k}$ in the dynamic Programming phase requires $N \times K$ computations of $\min_{j \in [\![1,i]\!]} C_{j-1,k-1} + c_{j,i}$, which are in $O(N)$, the complexity of this phase is in $O(K.N^2)$ which is a $O(N^3)$ as $K < N$. The final backtracking phase requires K computations having a complexity in $O(N)$, the complexity is in $O(K.N)$. It proves that the complexity of Algorithm 1 is in $O(N^3)$, the phase giving this complexity being the computation of matrix $c_{i,i'}$.

6 Discussions

This section discusses the implications and applications of Theorem 1 and Algorithm 1.

6.1 Relations with the state-of-the-art

The k-means problem was proven NP-hard in a Euclidean space of dimension 2 since [17]. This emphasizes that the hypothesis of non dominated solutions is crucial in this result. We note similarities with the 1-dimensional case, proven polynomially solvable thanks to a dynamic programming algorithm in [27] improved in [11]. Actually, Proposition 2 induces a similarity with the 1-dimensional case, the total order induces a 1-dimensional structure. The general 1-dimensional k-means problem is included in the 2 dimensional case in a Pareto Front, it is equivalent to cluster among a linear Pareto front. The general case without linearity of the Pareto front induces more complications, with no additivity of distance but a triangular inequality. Lastly, we note that an equivalent of the Conjecture 1 was proven for the p-median and p-centers problems in [8], leading to polynomially proven algorithms.

6.2 Improving the complexity of Algorithm 1?

In the Algorithm 1, the complexity is due to the initial computations of the matrix $c_{i,i'}$, whereas main part of the algorithm, the dynamic programming phase, has a complexity in $O(K.N^2)$. It is possible to reduce the number of computations noticing that some computations of $c_{i,i'}$ are not required. Indeed, in the computations $C_{i,k} = \min_{j \in [\![1,i]\!]} C_{j-1,k-1} + c_{j,i}$, the computation of $c_{i,j}$ may be useless if $C_{j-1,k-1} + \tilde{c}_{i,j}$ is higher than the current best value in the minimization, where $\tilde{c}_{i,j}$ is a lower bound of $c_{i,j}$ easier to compute. This opens numerical perspectives to accelerate the Algorithm 1.

6.3 Adding cardinality constraints

Similarly to [20], Algorithm 3 allows to incorporate cardinality constraints, considering only clusters $\mathcal{C}_{i,i'}$ with specific cardinality of $i' - i$. A first reason is that when $i' - i$ close to N, there are few chance to have very unbalanced clusters at optimality. A second reason could be to impose for representativity to avoid too small clusters. It can be a constraint to impose that the cardinal of the selected clusters must be close to $\frac{N}{K}$. In Algorithm 3, a first way to deal with such constraints is to set values $c_{i,i'} = +\infty$ for the $i < i'$ with the unwilled cardinality.

We note that such cardinality constraints can have a positive impact on the complexity of Algorithm 1. Computations $C_{i,k} = \min_{j \in [\![1,i]\!]} C_{j-1,k-1} + c_{j,i}$ are easier with less cases in j to enumerate. For the both continuous and discrete p-center problems, the computations of $c_{i,i'}$ are independent, the useless computations of $c_{i,i'}$ can be removed. Allowing for each $i \in [\![1, N]\!]$ only $\alpha.K$ definite values of $c_{i,i'}$, it improves the final complexity. A natural case would be to consider only the subsets $[\![i, i']\!]$ with $\lfloor N/K \rfloor - \alpha K < |i' - i| < \lfloor N/K \rfloor + \alpha k$ for a given $\alpha \in \mathbb{N}$. With such choices, the construction of the matrix $C_{i,k}$ is in $O(K^2.N)$ with operations $C_{i,k} = \min_{j \in [\![1,i]\!]} C_{j-1,k-1} + c_{j,i}$ having a complexity in $O(K)$. With such choices, the construction of the matrix $C_{i,k}$ is in $O(K^2.N)$.

6.4 Accelerated matheuristic derived from Algorithm 1

The complexity in $O(N^3)$ can be a bottleneck to compute Algorithm 1 for high values of N. $O(K.N^2)$, the complexity of the dynamic programming algorithm without the computations of $c_{i,i'}$, can be a more reasonable complexity.

A first possibility is to compute in $O(K)$ good lower bounds of $c_{i,i'}$ and use the dynamic programming with these estimations to compute partitions that are finally evaluated using the exact formula. As is, the dynamic programming has a complexity in $O(K.N^2)$.

Another possibility, similar to variable fixing heuristics as in [10], is to restrict the computations forbidding some partitions. Setting $c_{i,i'} = +\infty$ forbids the cluster $\mathcal{C}_{i,i'}$. With such choices, the construction of the matrix $C_{i,k}$ is in $O(K^2.N)$. Using infinite values of $c_{i,i'}$ is also a way to add a cardinality constraints for the clusters.

These two constructive matheuristics can be followed by local search iterations using the Lloyd's algorithm [16]. This steepest descent procedure can be considered only on the extreme points of the clusters with Proposition 1, leading to a complexity in $O(K)$ computations for each steepest descent iteration.

6.5 Towards a distributed implementation

The numerical computations of Algorithm 1 can be accelerated with a distributed implementation using the Message Passing Interface (MPI) framework, as in [18]. The computation of $c_{i,i'}$ are independent and are thus easily easy parallelized. The construction of the matrix $C_{i,k}$ requires independent computations for a given k, using the final values in $k-1$. For a parallel implementation sharing the memory, this requires just to wait that all the coefficient $C_{i,k-1}$ are terminated to start the computations of $C_{i,k-1}$. With a distributed implementation with MPI, this requires for each thread to broadcast the results of $C_{i,k}$ computations to the other threads.

6.6 Using the Algorithm 1 in higher dimensions?

The applicative motivation could concern multi-optimization preoccupations with three or more objectives. The bi-objective case is a first step. The 2-dimensional case is very specific thanks to the Proposition 2. However, Algorithm 1 can be used thanks to the reduction of dimension with Principal component analysis (PCA) or using the Johnson-Lindenstrauss lemma [6].

6.7 Applications to multi-objective optimization

The hypothesis defining E is verified for non-dominated points of bi-objective optimization. As stated in the introduction, the initial motivation of this work is to aid the decision makers when a multi-objective optimization approach without preference furnishes a large set of non dominated solutions. In this application, the value of K is small, for human analyses to give some preferences.

A posteriori, the complexity of Algorithm 3 allows also to consider these clustering algorithms inside multi-objective optimization meta-heuristics. Archiving Pareto fronts is a common issue of population meta-heuristics facing multi-objective optimization problems [25]. A key issue is to have diversified points of the Pareto front in the archive, to compute diversified solutions along the current Pareto front. Algorithm 3 can be used to address this issue, embedded in multi-objective optimization approaches. More specifically, clustering Pareto sets has an application for the diversification of genetic algorithms to select diversified solutions for cross-over and mutation phases [28]. For swarm particle optimization, clustering algorithm are also useful in the low-level implementation as shown in [22]. This applies also for the large class of multi-objective meta-heuristics [26].

Embedded in multi-objective meta-heuristics, Using Algorithm 3 would be called iteratively. Having a dynamic programming algorithm, this makes easier online optimization where several points are not changing from an iteration to another. In this case, the computation of matrix c can reuse the previous values that are still valid. Coefficients $C_{i,k}$ can also be computed quicker with previous computations.

6.8 Applications to optimization under uncertainty

Clustering non-dominated vectors has also applications in the context of optimization under uncertainty. It is a natural and common idea to model the uncertain data by the use of discrete scenarios for stochastic and robust optimization [23, 15]. A computational limitation can be the maximal number of scenario to incorporate in the optimization models, or the number of scenario to explore in meta-heuristics like in [13]. It induces a need for an algorithm to select a subset of representative scenarios to explore, and/or to aggregate the closest scenarios. In both cases, the problem is similar to select a given number of scenarios, maximizing the representativity of the selected scenarios The points to cluster represent in this case scenarios, the dimension of the space $\mathbb{R}^n$ is the number of uncertain parameters n.

In the case of robust optimization with discrete scenarios, worst-case analysis induce to consider only the Pareto non-dominated scenarios. The mid-point heuristic, aggregating non-dominated scenarios, is used for approximation results but also for the quality of primal heuristics in [4]. Partial aggregation following representative clusters of scenario is likely to improve the mid-point primal heuristic.

In the case of stochastic optimization, the progressive hedge algorithm uses scenario aggregation to derive primal heuristics [23, 13]. We note that dual heuristics in [9] aggregates also scenarios following partitions to have dual bounds for stochastic problems. The quality of the clustering impacts the quality of the heuristics in both cases, this makes sense to be careful on the clustering algorithms to use.

References

1. D. Aloise, A. Deshpande, P. Hansen, and P. Popat. NP-hardness of Euclidean sum-of-squares clustering. *Machine learning*, 75(2):245–248, 2009.
2. D. Arthur and S. Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 1027–1035. Society for Industrial and Applied Mathematics, 2007.
3. P. Awasthi, A. Blum, and O. Sheffet. Stability yields a PTAS for k-median and k-means clustering. In *51st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 309–318, 2010.
4. A. Chassein and M. Goerigk. On scenario aggregation to approximate robust combinatorial optimization problems. *Optimization Letters*, pages 1–11, 2017.
5. S. Dasgupta. *The hardness of k-means clustering.* Department of Computer Science and Engineering, University of California, San Diego, 2008.
6. S. Dasgupta and A. Gupta. An elementary proof of a theorem of Johnson and Lindenstrauss. *Random Structures & Algorithms*, 22(1):60–65, 2003.
7. N. Dupin. *Modélisation et résolution de grands problèmes stochastiques combinatoires: application à la gestion de production d'électricité.* PhD thesis, Ph. D. thesis, Lille 1, 2015.
8. N. Dupin and E. Talbi. Clustering in a 2-dimensional Pareto Front: the p-median and p-center problems are polynomially solvable. *arXiv preprint arXiv:1806.02098*, 2018.

9. N. Dupin and E. Talbi. Dual heuristics and new dual bounds to schedule the maintenances of nuclear power plants. *arXiv preprint arXiv:1806.00445*, 2018.
10. N. Dupin and E. Talbi. Parallel matheuristics for the discrete unit commitment problem with min-stop ramping constraints. *International Transactions in Operations Research*, pages 1–25, 2018.
11. A. Grønlund, K. Larsen, A. Mathiasen, J. Nielsen, S. Schneider, and M. Song. Fast exact k-means, k-medians and Bregman divergence clustering in 1d. *arXiv preprint arXiv:1701.07204*, 2017.
12. J.A. Hartigan and M.A. Wong. Algorithm AS 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108, 1979.
13. K. Haugen, A. Løkketangen, and D. Woodruff. Progressive hedging as a meta-heuristic applied to stochastic lot-sizing. *European Journal of Operational Research*, 132(1):116–122, 2001.
14. A. Jain. Data clustering: 50 years beyond k-means. *Pattern recognition letters*, 31(8):651–666, 2010.
15. A. Kasperski and P. Zieliński. Robust discrete optimization under discrete and interval uncertainty: A survey. In *Robustness Analysis in Decision Aiding, Optimization, and Analytics*, pages 113–143. Springer, 2016.
16. S. Lloyd. Least squares quantization in PCM. *IEEE transactions on information theory*, 28(2):129–137, 1982.
17. M. Mahajan, P. Nimbhorkar, and K. Varadarajan. The planar k-means problem is NP-hard. *Theoretical Computer Science*, 442:13–21, 2012.
18. F. Nielsen. *Introduction to HPC with MPI for Data Science.* Springer, 2016.
19. F. Nielsen and R. Nock. Further heuristics for k-means: The merge-and-split heuristic and the (k, l)-means. *arXiv preprint arXiv:1406.6314*, 2014.
20. F. Nielsen and R. Nock. Optimal interval clustering: Application to Bregman clustering and statistical mixture learning. *IEEE Signal Processing Letters*, 21(10):1289–1292, 2014.
21. T. Peugeot, N. Dupin, M-J Sembely, and C. Dubecq. MBSE, PLM, MIP and Robust Optimization for System of Systems Management, Application to SCCOA French Air Defense Program. In *Complex Systems Design & Management*, pages 29–40. Springer, 2017.
22. G. Pulido and C. Coello. Using clustering techniques to improve the performance of a multi-objective particle swarm optimizer. In *Genetic and Evolutionary Computation Conference*, pages 225–237. Springer, 2004.
23. R. Rockafellar and R. Wets. Scenarios and policy aggregation in optimization under uncertainty. *Mathematics of operations research*, 16(1):119–147, 1991.
24. N. Slonim, E. Aharoni, and K. Crammer. Hartigan's K-Means Versus Lloyd's K-Means-Is It Time for a Change? In *IJCAI*, pages 1677–1684, 2013.
25. E. Talbi. *Metaheuristics: from design to implementation*, volume 74. John Wiley & Sons, 2009.
26. E-G. Talbi, M. Basseur, A. Nebro, and E. Alba. Multi-objective optimization using metaheuristics: non-standard algorithms. *International Transactions in Operational Research*, 19(1-2):283–305, 2012.
27. H. Wang and M. Song. Ckmeans. 1d. dp: optimal k-means clustering in one dimension by dynamic programming. *The R journal*, 3(2):29, 2011.
28. E. Zio and R. Bazzo. A clustering procedure for reducing the number of representative solutions in the pareto front of multiobjective optimization problems. *European Journal of Operational Research*, 210(3):624–634, 2011.

The Evaluation-times Constrained Optimization (ECO) Problem and Its General Solver Model

Kenichi Tamura

Tokyo Metropolitan University, 1-1 Minamiosawa, Hachioji, Tokyo JAPAN
tamu@tmu.ac.jp

1 Introduction

In recent years, as various engineering technologies have progressed, applications of optimization techniques have diversified, as shown in Fig. 1. Cases 2 and 3 in the figure especially tend to have black-box and expensive objective functions regarding time and/or money for each evaluation. Meanwhile, in the real world, time and/or money constraints are usually imposed beforehand on any engineering development project. Therefore, practitioners who handle such expensive black-box objective functions must find better approximated solutions to as great a degree as possible, under the evaluation-times constraint induced by the time and/or money constraints. We refer to the general class of such problems as the *Evaluation-times Constrained Optimization (ECO)* problem.

Let us consider an example in case 2 to understand the importance of the ECO problem. Suppose that an objective function value is calculated based on a solution from an expensive simulator that takes 1 day to output one solution. Furthermore, assume that a practitioner is given a time constraint of 30 days to attempt to optimize this expensive objective function. In this case, the practitioner can run the simulator a maximum of 30 times, which means that the practitioner has to find a good approximated solution in 30 evaluation-times for the objective function. However, typical optimization algorithms would not work well for such a case because they are not configured based on the constraint information. They cannot adapt to the constraints originated from the real world.

In this short paper, we formulate ECO problems and then provide a general solver model for them. The general solver model is a population search model designed to carry out a strategy that gradually shifts from exploration to exploitation as the evaluation-times come close to the constraint. Specifically, a parameter set of the model, affecting the population behavior, is consecutively adjusted every iteration to execute the strategy. Following this general model, we would create new population search algorithms for ECO problems or customize existing population search algorithms, such as particle swarm optimization (PSO) [1], differential evolution (DE) [2], cuckoo search (CS) [3], and spiral optimization (SPO) [4], for ECO problems.

2 Formulation of Evaluation-times Constrained Optimization (ECO) Problems

This section aims to formulate the ECO problems.

Defining an expensive objective function $f : \mathbb{R}^D \to \mathbb{R}$, the number of evaluation-times t, and a given evaluation-times constraint T, we can formally describe ECO problems as follows.

$$\underset{\boldsymbol{x} \in \mathbb{R}^D}{\text{Decrease}}\, f(\boldsymbol{x}) \text{ a.m.a.p. subject to } t \leq T.$$

This uses "Decrease" and "a.m.a.p. = as much as possible" to attenuate the usual command "Minimize" because it is generally impossible in this scenario to guarantee that the algorithm will strictly minimize and find an optimal solution for any T. In fact, typical optimization theories assume infinite evaluation-times.

Furthermore, this formalization can be more specific at the expense of generality. Suppose $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_t\}$ are sequential solutions until iteration t by an algorithm for which one iteration means one function evaluation. In this case, the ECO problem can be represented as follows.

$$\underset{\boldsymbol{x}_t \in \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}}{\text{Decrease}} \min\{f(\boldsymbol{x}_t)\ (t = 1, \ldots, T)\ \} \text{ a.m.a.p.}$$

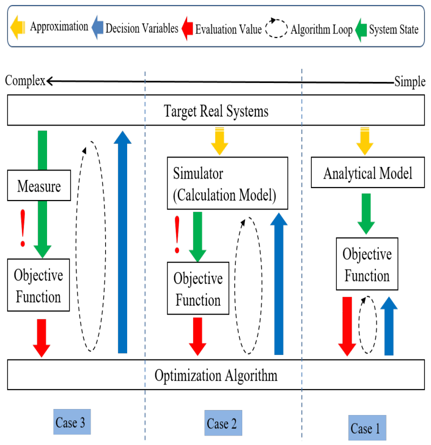


Fig. 1 Optimization Application Classes
(An exclamation mark indicates that it is expensive)

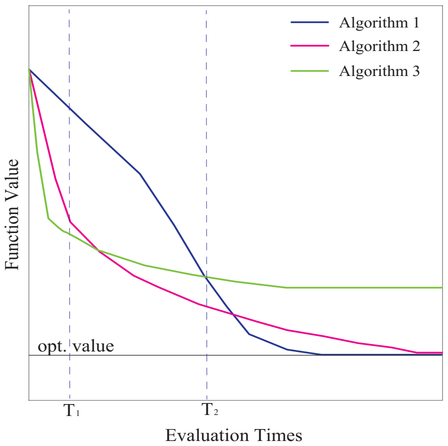


Fig. 2 Algorithm Evaluation for ECO Problems
(Minimization Case)

Note that a similar and simpler expression is “Decrease$_{\boldsymbol{x}_T}$ $f(\boldsymbol{x}_T)$ a.m.a.p.” and that “Decrease” can be replaced with “Increase” if a maximization problem is considered.

3 A General Solver Model for ECO Problems

This section aims to suggest a general solver model for ECO problems.

First, we consider the following general population search model for common optimization problems for an objective function f.

$$\boldsymbol{X}_{k+1} = \mathcal{A}(\boldsymbol{X}_{1:k}, \boldsymbol{F}_{1:k}, \boldsymbol{P}) \quad (k = 1, \ldots, K) \tag{1}$$

where $\boldsymbol{X}_k := \{\boldsymbol{x}_{1,k}, \ldots, \boldsymbol{x}_{M,k}\}$ is a population of M search points at an iteration k, $\boldsymbol{F}_k := \{f(\boldsymbol{x}_{1,k}), \ldots, f(\boldsymbol{x}_{M,k})\}$, $\boldsymbol{X}_{1:k} := \{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_k\}$, $\boldsymbol{F}_{1:k} := \{\boldsymbol{F}_1, \ldots, \boldsymbol{F}_k\}$, $\boldsymbol{P}$ is a parameter set that affects the behavior of the population (interaction among the population) for the next step, and K is the maximum of the iteration. This general model means that the next population is determined based on the search history data and the parameter set. This model can represent almost any population search algorithm such as PSO [1], DE [2], CS [3], and SPO [4].

The performances of typical optimization algorithms are measured by each evaluation-times when each objective value sequences almost reach the optimal value. However, ECO algorithms must be measured by the best objective value when the evaluation-times reaches the given constraint. For example, in Fig.2, Algorithm 1 is the most efficient in the context of typical minimization. However, when the evaluation-times constraints are T_1 and T_2, the most efficient algorithms are Algorithm 3 and Algorithm 2, respectively. This indicates that the evaluation-times constraint is essential information for algorithms performance. Therefore, the algorithms for ECO problems should utilize the given evaluation-times constraint T in the search mechanisms.

Based on this principle, we here consider adjusting the parameter set $\boldsymbol{P}$ of the usual model (1) using the constraint T to make the population $\boldsymbol{X}_k$ behave appropriately for ECO problems. To this end, we focus on a well-known effective strategy in metaheuristics that finds an appropriate balance between *exploration* and *exploitation* for a target problem. Note that exploration is a strategy that globally searches, and exploitation is a strategy that intensively searches around good solutions. Here, as such an appropriate balance for ECO problems, we suggest gradually shifting exploration to exploitation as the evaluation-times t approaches T. It is noted that, in this model, the evaluation-times t is defined by $t = M \times k$. Thus, for a given evaluation-times constraint T, the population size M and the maximum iteration K must be set to satisfy $K = T/M$.

To realize this concept, we introduce a measure $E(\boldsymbol{X}_{k+1})$, evaluating a relation between all points $\boldsymbol{x}_{i,k+1}$ $(i = 1, \ldots, M)$ and the current best point $\boldsymbol{x}_{\text{best}}$, and a target trajectory $R(k;T)$ of $E(\boldsymbol{X}_{k+1})$ which gradually decreases as $k \to K = T/M$ (i.e., $t \to T$). The measure $E(\boldsymbol{X}_{k+1})$ can evaluate the degree of balance between exploration and exploitation for the current population. Thus, when $E(\boldsymbol{X}_{k+1})$ is decreasing, we can judge that the exploitation is getting stronger. The

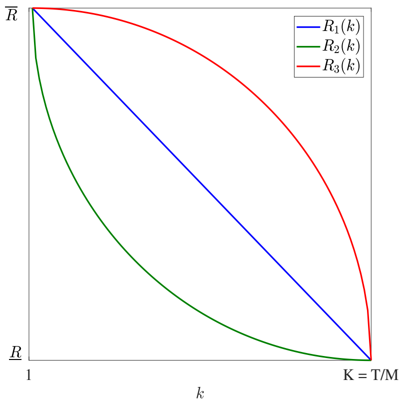


Fig.3 Examples of $R(k, T)$

```
Algorithm (Proposed General Solver Model for ECO Problems)
0) The evaluation-times constraint $T$ is given. Set the number
   of search points $M$ and the maximum iteration number $K$
   to satisfy $T = M \times K$.
1) Set $k = 1$. Give the initial population $\boldsymbol{X}_1$ and evaluate the
   objective values $\boldsymbol{F}_1$
2) If $k = K(= T/M)$, then terminate and output the solution
   with the best objective value. Otherwise go to 3).
3) Update the parameter set $\boldsymbol{P}_{k+1} = \text{minimize}_{\boldsymbol{P}}(R(k;T) - E(\boldsymbol{X}_{k+1}))^2$.
4) Update the search population $\boldsymbol{X}_{k+1} = \mathcal{A}(\boldsymbol{X}_{1:k}, \boldsymbol{F}_{1:k}, \boldsymbol{P}_{k+1})$
   and evaluate the objective values $\boldsymbol{F}_{k+1}$.
5) Set $k := k + 1$ and go back to 2).
```

Fig.4 Proposed General Solver Model for ECO Problems

converse is also true. The target trajectory $R(k;T)$ provides an ideal decreasing balance from exploration to exploitation that $E(\boldsymbol{X}_{k+1})$ should follow as $k \to K = T/M$ (i.e., $t \to T$). As examples of $R(k;T)$, we provide the following three decreasing functions:

- $R_1(k;T) = (\underline{R} - \overline{R})(k-1)/(K-1) + \overline{R}$,
- $R_2(k;T) = (\underline{R} - \overline{R})\sqrt{1 - (k-K)^2/(1-K)^2} + \overline{R}$,
- $R_3(k;T) = (\underline{R} - \overline{R})(-\sqrt{1 - (k-1)^2/(K-1)^2} + 1) + \overline{R}$,

where $K = T/M$ and $\overline{R}$ is usually set to $E(\boldsymbol{X}_1)$. These trajectories are as shown in Fig. 3. From the usual model (1), $E(\boldsymbol{X}_{k+1}) = E(\mathcal{A}(\boldsymbol{X}_{1:k}, \boldsymbol{F}_{1:k}, \boldsymbol{P}))$ can be regarded as a function of the parameter set $\boldsymbol{P}$. Thus, we can expect to adjust the parameter set $\boldsymbol{P}$ to make $E(\boldsymbol{X}_{k+1})$ follow the ideal trajectory $R(k;T)$. Specifically, we propose adjusting $\boldsymbol{P}$ every iteration by solving the following problem.

$$\boldsymbol{P}_{k+1} = \underset{\boldsymbol{P}}{\text{minimize}}(R(k;T) - E(\boldsymbol{X}_{k+1}))^2 \quad (k = 1, \ldots, K). \tag{2}$$

This method determines the next parameter set $\boldsymbol{P}_{k+1}$ to make $E(\boldsymbol{X}_{k+1})$ follow $R(k;T)$ for each iteration k. If the number of elements of $\boldsymbol{P}$ is small, this problem could be easily solved. We can describe the proposed general solver model as shown in Fig.4.

4 Conclusions

This short paper has formulated a new class of black-box optimization problems, called the Evaluation-times Constrained Optimization (ECO) problem, in which the evaluation-times of the objective function are constrained beforehand. Furthermore we showed a general solver model for ECO problems that is designed to adapt a typical population search model to ECO problems. Our future work is to use this proposed general model to develop new population algorithms for ECO problems or adapt existing population algorithms to ECO problems.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Number 16K18121.

References

1. J. Kennedy and R. C. Eberhart, Particle swarm optimization, Proc. IEEE Int. Conf. Neural Netw., pp. 1942-1948 (1995).
2. R. M. Storn and K. V. Price, Differential Evolution - A Simple and Efficient Heuristic for Global Optimization over Continuous Space, Journal of Global Optimization, vol. 11, pp. 341–359 (1997).
3. X. S. Yang and S. Deb, “Cuckoo Search via Levy Flights,” Proc. of World Congress on Nature & Biologically Inspired Computing (NaBIC2009), pp. 210-214 (2009).
4. K. Tamura and K. Yasuda, “The Spiral Optimization Algorithm: Convergence Conditions and Settings,” IEEE Trans. Systems, Man, and Cybernetics: Systems, vol. PP, no. 99, pp. 1–16 (2017).

A pickup and delivery problem with multi-trips, multi-flux, multi-vehicles and break placement

A. M. Noumbissi Tchoupo[1], A. Yalaoui[1], L. Amodeo[1], F. Yalaoui[1], E. Thivet[2], P. Flori[2]

[1] University of Technology of troyes, Lab-LOSI, ICD-UMR-CNRS 6281, Troyes, France
{*moise.noumbissi_tchoupo*,alice.yalaoui, lionel.amodeo, farouk.yalaoui}@utt.fr
[2] Hospital Centre of Troyes , 101 avenue anatole france, Troyes, France
{eric.thivet,pauline.flori}@ch-troyes.fr

1 Introduction

This paper addresses a new Fleet size and mix Pickup and Delivery Problem with multi-flux, compatibility constraints and scheduling, encountered in real-world logistics. In this problem, there are a team of drivers and multiple vehicle types available to cover a set of pickup and delivery's demand pair. Each demand has a time windows, a number of product type's trolleys. The demands are paired such that a pickup demand is coupled with a unique delivery demand. Each pair of demands is carried by a driver with the same vehicle, and the pickup demand must be performed before its delivery demand. Demands and vehicle types must satisfy a set of compatibility constraints that specify which demands cannot be covered by which vehicle types and which requests cannot be shipped together. Each schedule of a driver must satisfy the daily driver's work rules. In this problem, a driver is allowed to change the vehicle in it's schedule. The total cost of a solution is the sum of the total fixed cost of driver's number, the total fixed cost of used vehicles and the total mileage cost.

Recently, many extensions of pickup and delivery problems motivated by real life problems were discussed in the literature. [4] proposed a set partitioning formulation to solve a practical heterogeneous fleet pickup and delivery problem with unloading sequence constraints. They also took into account practical aspects such as: multiple time windows, multiple depots, compatibility constraints between carrier and vehicles, orders and vehicles, and between orders. [2] proposed a branch and price algorithm to tackle pickup and delivery problems with soft time windows. Most of the complications involved in the pickup and delivery problems such as time windows, capacity, compatibility constraints, and driver's works rules have not been addressed together in the literature. To solve this new pickup and delivery problem, in the next section an effective hybrid ant colony optimization algorithm is proposed to solve this problem.

2 Solution methodology and results

The ant colony optimization algorithm is chosen because it proved its effectiveness in vehicle routing problems ([1,3]) since its constructive method does not require a repair procedure. To present the algorithm, firstly the procedure of a solution construction by an ant is given, secondly the updating formulae of the pheromone are shown. The dedicated local search algorithms which are proposed to improve accuracy of the ant colony algorithm, are described in the next subsections.

2.1 Construction of a solution and pheromone updating procedure

A solution is composed of a set of schedule, characterised by a suit of trips. An ant, using pheromone, constructs the schedules sequentially. It opens a new schedule for a new driver if among the remaining nodes to visit any of them can be inserted in the current schedule. A new schedule is initialized with the demand chosen to be completed using a vehicle type with the larger area, which respect the compatibility constraints, when in the construction of a solution the request found using pheromone and informations in the current trip, cannot be inserted in the current trip without violate the driver's rules.

Given a partial scheduling s which is building, we consider r the last trip constructed. Let p_i the last pickup demand satisfying in the trip r. The next request j to insert is chosen to be randomly inserted in the current trip using probability:

$$P_{ij}^{r} = \frac{\tau_{ij}^{\alpha_1}\eta_{ij}}{\sum_{l\in S_i^r}\tau_{il}^{\alpha_1}\eta_{il}}\text{if } j \in S_i^r\text{; and 0 otherwise}$$

τ_{ij} denotes the trail of pheromone on arc (i,j). S_i^r is the set of eligible requests which can be performed after request i (with p_i the last pickup demand complete in the last trip r). The parameters α_2 and η_{ij} modulated consecutively the importance between the visibility and the pheromone, and the visibility value used to guide the ant.

$$\eta_{ij} = \frac{1}{l_j + \alpha_2 d_{p_i p_j}} \tag{1}$$

with $\alpha_2 > 0$ a positive fixed scalar, $d_{p_i p_j}$ the distance from the location of p_i to p_j, and l_j the latest time at which the service may begin at node p_j .

The initialization and pheromone updating procedures are important features to enhance promising arcs and then explain the convergence of the algorithm. At the beginning of ACO, pheromones are initialized as follow:

$$\tau_{ij} = \tau_0 \text{ if } (i,j) \in E\text{; and 0 otherwise}$$

with τ_0 a fixed scalar.

When all ants have constructed their solutions, the trail pheromone deposited on all edges are updated by the formulae:

$$\tau_{ij} \leftarrow \begin{cases} \rho\tau_{ij} + (1-\rho)\Delta_{ij}^{*} & \text{if } p_i \text{ precedes } p_j \text{ in the best ant} \\ \rho\tau_{ij} & \text{otherwise} \end{cases} \tag{2}$$

$$\text{with } \Delta_{ij}^{*} = \frac{(\text{number of demands in } s_i^{*})^{\alpha_3}}{\text{total cost of the best solution found}}.$$

And s_i^* the schedule which contains request i in the best ant, ρ ∈[0;1] fixed and α_3 positive fixed scalars.

The idea of using Δ_{ij}^{*} is to enforce the construction of trips with a good packing. This formulae to update pheromone allowed the algorithm to converge to a unique solution.

2.2 Local search algorithms

This section describes five local search algorithms used to improve a solution constructed by an ant. Each local search algorithm is designed to handle a specific aspect of objective function (the sum of fixed cost of drivers, sum of fixed cost of vehicles used and the routing cost).

Heuristic $H1$ is used to decrease the sum of the number of drivers used. $H1$ tries to remove each schedule of the solution. So at each iteration, a schedule is selected and remove from the solution. Thus, the requests presents of this schedule are reinserted in the partial solution. Every request is inserted in the partial solution at the positions (pickup and delivery positions) that minimizing the total cost. If finally, all the requests presented in the trip r are reinserted, the solution is updated if the new solution has a better cost than the previous one.

The local search algorithm $H2$ is proposed to decrease the sum of fixed cost of the vehicles used. Given a solution, for each schedule and for each trip in this schedule, the algorithm finds the type of the vehicle able to performed the trip at the cheapest cost. At each iteration of the heuristic $H3$, a request is randomly chosen and removed from the solution, and, the pickup and delivery demands of this request are reinserted at the positions that minimizes the total cost. The process is repeat for a given number of iterations.

The algorithm $H4$ improves a given solution by optimizing each trip of this solution. To optimize a trip, an insertion order of requests presents in this trip is randomly generated. In the order of completion, these requests are inserted in a new trip (each insertion of the request is done such that the cost after insertion of the request is minimized). If all requests are inserted in the new trip with a better cost, the trip is replaced by the new one.

The algorithm $H5$ at each iteration, removes a trip in a schedule with more than one trip, and tries to reinsert the requests (from the removed trip), in the partial solution. If all the requests are reinserted, a new solution is obtained. And the previous solution is replaced by the new one, if it has a better cost.

The pseudo code of hybrid ACO algorithm used is given by Algorithm 1. In this pseudo code, the algorithms $H1$, $H2$, $H3$, $H4$ and $H5$ at lines 6, 8 and 10 are mainly used sequentially to decrease the total cost of the solution constructed by an ant. The choice of the order of using of local search algorithms was made after several experiments to optimize the algorithm performances.

Algorithm 1 Pseudo code of HACS algorithm

```
Initialization of parameters
while the best solution is not improved in a given number of iterations do
    for each ant of the population do
        Construct a solution to complete all demands
        while we decrease the number of drivers do
            Apply respectively algorithms: H1, H3, H4 and H5
        end while
        Apply algorithm H2
        while we decrease the total cost of the current solution do
            Apply respectively algorithms: H1, H3, H4, H5 and H2
        end while
        Apply the local updating pheromone
    end for
    Apply the global updating pheromone
end while
```

2.3 Computational experiments

The algorithms have been implemented on eclipse, the programming language was Java and the experiments have been carried out on a 2.6 GHz and 8 Go of RAM. Based on well know Solomon's Benchmark for vehicle routing, a benchmark with reasonable size (up to 100 demands) is generated.

There are six parameters that can significantly affect the performance of the hybrid ACO: $\rho = 0.85$, $\alpha_1 = 3$, $\alpha_2 = 0$, $\alpha_3 = 3$, $\tau_0 = 1$, number of ant $= 2N$. Where N is the number of demands. These values are found by the Taguchi experimental design.

The proposed algorithm finds a feasible solution for each instance of the proposed benchmark. These solutions provide a first set of best know solutions for the future methods of resolution.

Experiments on the Hospital Centre of Troyes benchmark (consisting of instances with 100 demands) are also realized. The experiments show that on average, the result obtained by the proposed algorithm decreases in average 26% of the cost of the solution for each day of the week. The computational time spent by the hybrid ant colony algorithm on solving each instance is just over two minutes.

3 Conclusion

In this paper, we have presented a pickup and delivery problem which involves a set of practical issues such as time windows, heterogeneous vehicle types, DOT rules, and compatibility constraints. These issues are commonly seen in real world logistics operations, but have received little attention in the literature. In this paper, it is designed an ant colony algorithm to solve the problem. The proposed ant colony algorithm is combined with dedicated local search algorithms to improve the quality of the solutions obtained. To test the proposed hybrid ACO algorithm, 56 instances up to 100 demands based on well-know Solomon's benchmark were introduced. The tests on the industrial benchmark show the effectiveness of our approach. In the future, it would be interesting to compare our approach with other algorithms.

References

1. Farah Belmecheri, Christian Prins, Farouk Yalaoui, and Lionel Amodeo. An ant colony optimization algorithm for a vehicle routing problem with heterogeneous fleet, mixed backhauls, and time windows. *IFAC Proceedings Volumes*, 42(4):1550–1555, 2009.
2. Andrea Bettinelli, Alberto Ceselli, and Giovanni Righini. A branch-and-price algorithm for the multi-depot heterogeneous-fleet pickup and delivery problem with soft time windows. *Mathematical Programming Computation*, 6(2):171–197, 2014.
3. M Noumbissi Tchoupo, A Yalaoui, L Amodeo, F Yalaoui, and F Lutz. Ant colony optimization algorithm for pickup and delivery problem with time windows. In *International Conference on Optimization and Decision Science*, pages 181–191. Springer, 2017.
4. Hang Xu, Zhi-Long Chen, Srinivas Rajagopal, and Sundar Arunapuram. Solving a practical pickup and delivery problem. *Transportation science*, 37(3):347–364, 2003.

Iterated Local Search for the Integrated Single Item Lot Sizing Problem for a Flow Shop Configuration With Energy Constraints

Melek Rodoplu, Taha Arbaoui, and Alice Yalaoui

ICD, LOSI, University of Technology of Troyes, UMR 6281, CNRS, Troyes, France
{melek.rodoplu, taha.arbaoui, alice.yalaoui}@utt.fr

1 Introduction

Recently, the increase of the use of renewable energies and the need for more energy efficiency are one of the most important concerns of the governments. Improving the energy efficiency and incrementing the renewable energy consumption in the industrial sector which has the largest share of energy consumption become one of the most effective steps to reach defined *"20-20-20 Energy Targets"* of European Union. Therefore, especially in the last few decades, energy-efficiency-based studies received a considerable attention from both researchers and companies. With the increasing awareness for energy efficient production systems, the studies which are conducted for classical lot sizing and scheduling problems have evolved to energy-efficiency based lot sizing and scheduling studies. When the previous studies are examined, it is seen that the common approach applied to reduce the energy costs is to reduce the energy consumption. None of them take into account the energy supplier conditions and build a relation between the energy related constraints and production constraints. The work of [1] which introduces the energy aspect to the single item lot sizing problem can be referred as the first attempt which combines both energy supplier and producer side constraints. In our study, we improve the work of [1] and integrate single item lot sizing problem in flow shop systems with the energy contract selection problem.

Energy producers propose different energy tariffs and contract options to their customers to manage their energy generation strategies in a more efficient way. Selecting best energy option which can cover the need of the production system is another important decision which has to be made by manufacturers. The aim of our study is to identify optimum energy contract option and the optimum production sizes for each period and each machine for a given planning horizon. The renewable energy sources are also taken into account in the contract capacity selection process by aiming to generate more environmental friendly production systems.

As it is proved by [2], capacitated lot sizing problem is NP-hard problem. Therefore, it is quite hard to reach optimum solution of the problem within a reasonable time period. In this study, we propose to apply an Iterated Local Search approach to overcome this difficulty.

2 Problem Definition

A manufacturing system which is made of N machines and N buffers is considered. The whole planning horizon is divided into T periods. Each period is defined by its length (L_t), its external demand (d_t) to be satisfied, the electricity price (Co_t). The objective is to identify the production quantity to be produced and the optimum energy contract option by minimizing the production, holding, set-up and energy costs. Three types of energy sources (k) (traditional, solar ,wind) are considered. The contract options ($V_{k,l}$) proposed by the supplier are characterized by their subscription costs ($Cost_{k,l}$). In the case of excessively high or low consumption from the contacted capacity, exceeding portions are penalized with defined penalty cost ($\mho_k$). The penalty tolerance level is considered 10% in this study. The objective function of the proposed approach can be written as follows:

$$\text{Minz}=\sum_{t=1}^{T}\sum_{m=1}^{N}(\psi_{m,t}.x_{m,t}+h.I_{m,t}+w_{m,t}.y_{m,t})+\sum_{l=1}^{R_k}\sum_{k=1}^{K}(Cost_{k,l}.P_{k,l})$$
$$+\sum_{t=1}^{T}\sum_{k=1}^{K}(\mho_k.(AC_{k,t}+BC_{k,t}))$$

Table 1. Notations

The Parameters	Definition
$\psi_{m,t}$	Electrical consumption cost of machine m at period t
h :	Holding cost per unit
$w_{m,t}$:	Setup cost of machine m in period t
$Cost_{k,l}$:	Subscription cost for energy source k option l.
$\mho_k$:	Penalty cost for energy source k.
The Decision Variables	Definition
$x_{m,t}$:	Quantity produced on machine m in period t.
$I_{m,t}$:	Inventory level of machine m at the end of period t.
$y_{m,t}$:	A binary variable, equal to 1 if machine m is setup in period t,0 otherwise.
$P_{k,l}$:	A binary variable, equal to 1 if energy source k and option l is selected 1,0 otherwise.
$AC_{k,t}$:	The energy used above the contracted value for energy source k in period t.
$BC_{k,t}$:	The energy used below the contracted value for energy source k in period t.

In the first part of the objective function, electricity based production cost, holding cost and setup costs are minimized. The second part selects the optimum energy contract option which can cover the energy need of the manufacturing system by minimizing energy costs and the third part calculates the penalty costs that can arise from any deviation from contracted option values. The objective function is followed by the lot sizing, flow shop configuration and energy related constraints. We suggest readers to review the previous work of authors for the detailed mathematical model. [3]

3 Solution Approach

Iterated local search (ILS) is a single-solution based metaheuristic algorithm which consists of three basic components: an initial solution generation mechanism, an improvement procedure and a perturbation mechanism which is used to escape from local minimum [5]. The main steps of the solution approach are given in Algorithm 1. As it can be observed from the solution algorithm, ILS approach starts with an initial solution and a local search is performed on the initial solution. Since the set of decent moves are applied in the local search procedure, this case leads to trapping in the local minimum. To escape from the local minimum point, perturbation procedure is applied. Therefore, the current solution is moved to the new walking regions in the solution space. This local search and perturbation procedures are repeated until the stopping condition is met. The movements of the found solutions are illustrated on Figure 1.

```
$s_0$=GenerateInitialSolution
$s^*$=Local search ($s_0$)
repeat
    $s^{'}$=Perturbation($s^*$, history)
    $s^{*'}$=Local Search($s_{\prime}$)
    $s^*$=AcceptanceCriterion($s^*$, $s^{*'}$, history)
until termination condition met
```

Algorithm 1: Iterated Local Search Algorithm [4]

In our study, we generate *initial solution* by assigning production quantities to the machines in a random way by following formula:

$$x_{m,t}{=}d_t{+}random(0, x_{m-1} - d_t) \quad \forall m = 2, ..., N, \forall t = 1, ..., T$$

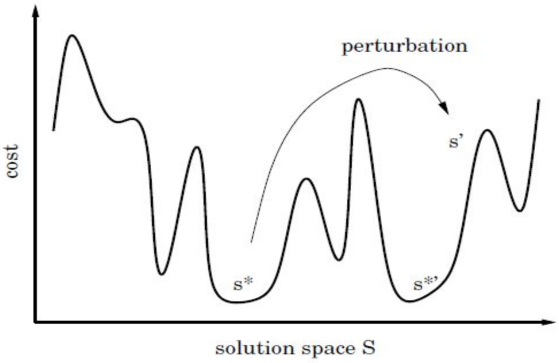


Fig. 1. Graphical representation of ILS solution approach [4]

This stipulates that the quantity on the last machine for the given period is the summation of the demand of the this period and a random quantity. The maximum value of this random quantity equals to difference between the quantity produced on the previous machine and the demand. Since the flow shop configuration is considered in this study, this approach ensures not to produce more than what is available from previous machines.

When it comes to identifying the movements in the solution space, we propose three moves: shifting the production inside of the periods, load balancing and reducing the inventory level.

Shifting the production may reduce the setup cost. Secondly, load balancing strategy aims to shift the starting time of the machines in a period to have a higher (or a lower) power usage within the period. Lastly, in reducing inventory strategy, the production quantities are shifted by keeping the same setup cost. Therefore, the machines run in the same periods but productions will change by shifting the production of some products from one period to another.

4 Conclusion

The main contribution of this study is to combine energy contract selection problem with the single item lot sizing problem for flow shop systems. It is aimed to built a harmony between production and energy decisions to contribute to the energy efficiency in the industry. The energy capacity options for renewable energy sources are taken into account, therefore, it is targeted to generate more environmental friendly production systems. Since the handled problem is NP-hard problem, an iterated local search approach is proposed to reach *"good"* results within a *"reasonable"* time period. The developed ILS facilitates to define optimum contract option covering the energy need of the manufacturing system and optimum production sizes which satisfy customer demand by overcoming the NP-hardness of the adressed problem. The first results of proposed approach present high quality solutions within an affordable computation time.

References

1. Masmoudi, O., Yalaoui,A.,Ouazene,Y.,Chehade,H.: Lot sizing in multi-stage flow line production system with energy consideration.International Journal of Production Research 55(2017)1640-1663.
2. Florian, M., Lenstra, J. K.,Rinnooy Kan, A. H. G.:Deterministic production planning: Algorithms and complexity. Management science, 26(1980), 669-679.
3. Rodoplu M., Arbaoui T., Yalaoui A., (2018). Energy Contract Optimization for the Single Item Lot Sizing Problem in a Flow-Shop Configuration and Multiple Energy Sources. Manuscript sumbitted for publication.
4. Loureno, H. R., Martin, O. C., Sttzle, T.:Iterated local search: Framework and applications. In Handbook of metaheuristics. Springer, Boston, MA.(2010),363-397.
5. Avci, M.,Topaloglu S.: A multi-start iterated local search algorithm for the generalized quadratic multiple knapsack problem. Computers & Operations Research 83 (2017),54-65.

Estimation-based algorithm for a stochastic one-commodity pick-up & delivery travelling salesman problem

S. Hadjadj and H. Kheddouci

Laboratoire d'informatique en image et systèmes d'information, Bâtiment Nautibus 43, bd du 11
Novembre 1918 69622 VILLEURBANNE CEDEX
mohamed-seddik.hadjadj@liris.cnrs.fr
hamamache.kheddouci@univ-lyon1.fr

1 Introduction

This work is carried out in collaboration with a company which specializes in the sale of ready-mix concrete.
Ready-mix concrete is normally delivered in *mixer trucks*. This type of truck is heavy, cumbersome, expensive, and can be disproportionate in some cases, especially when delivering small quantities of concrete.
Therefore, the company wants to propose a new delivery method using small containers (500 litre bins) to reduce delivery costs and deal more effectively with orders of small quantities.
This new method is a two-step process :

1. A vehicle delivers a number of bins of concrete to the customer ;
2. The next day, the vehicle returns to the customer to pick up empty bins.

To ensure the profitability of this method, the company needs a decision support system that can generate efficient **pick-up & delivery tours** taking into account vehicle capacity constraint and **recycling constraint**.
Indeed, if a bin is totally empty and clean when it is picked up from a customer, it could be directly supplied to another customer. Otherwise, it must be immediately routed to a recycling centre before it can be delivered again (the unconsumed concrete is then recycled and the bin cleaned). Knowing that the state of a bin is uncertain before the vehicle arrives at customer's location, the a priori planned vehicle route may changes during time to include recycling centre(s) whenever necessary. This uncertainty involves dealing with **stochastic vehicle routing**.
This paper aims to provide an efficient approach to build pick-up & delivery tours minimizing the loss of quality caused by potential detours to recycling centres.

2 Literature Review

We consider a stochastic One-Commodity pick-up & delivery travelling salesman problem.

2.1 Pick-up & delivery problems

There are three main classes of pick-up & delivery problem in the literature :

One-to-one problems One or more vehicle have to carry n commodities, where each commodity has a specific origin and destination. One of the best known examples of this class is the *Dial-a-Ride problem* which consists in transporting people from an origin to a destination. The problem has been studied for both single [4] and multiple [5] vehicle cases, with various types of constraints related to ride times, time windows [6, 7]...

One-to-many-to-one problems Commodities are divided into "delivery commodities" and "pick-up commodities". One or more vehicle have to carry the delivery commodities from the depot to the customers and the pick-up commodities from the customers to the depot. Assuming that n_p is a set of pick-up customers, and n_d a set of delivery customers, two cases have been distinguished for these problems : single demands, where $n_p \cap n_d = \emptyset$, and combined demands, where $n_p \cap n_d \neq \emptyset$. For the latter case, [8] consider various possible path types such as *Hamiltonian* path, where each customer is visited once such that pick-up and delivery are performed simultaneously, as well as *Double-path* where each customer that has a combined demand (pick-up and delivery) is visited twice, the first time for a pick-up, the second for a delivery. Several heuristics have been proposed for both path types for the single and the multi-vehicle cases [9, 10]...

Many-to-many problems One or more vehicle have to transport goods between customers knowing that each customer can be a source or a destination of any type of good. Among the problems of this class, the *One-Commodity pick-up and delivery travelling salesman problem* was introduced in [11]. A single vehicle with a known and finite capacity has to carry a single commodity between pick-up customers and delivery customers, a picked up commodity can be supplied to a delivery customer. This problem is known to be *NP-Hard*. Moreover, checking the existance of a feasible solution is an *NP-Complete* problem [13]. Studies on such problems are relatively scarce. A branch and cut algorithm has been proposed in [11] for small instances, and two heuristics have been developed in [12] to tackle larger instances, in particular by defining ”the infeasibility of a path”, and adapting the nearest neighbourhood heuristic to increase the chance of obtaining a feasible solution. Furthermore, [14] have proposed a hybrid method combining GRASP (greedy randomized adaptive search procedure) and VND (variable neighbourhood descent) metaheuristics. This method gave better results than the previously proposed ones.

For a detailed survey on pick-up and delivery problems, we refer the reader to [15].

2.2 Stochastic/Dynamic vehicle routing problems

Vehicle routing problems can be classified according to the information quality and evolution. Thus, an input information can be deterministic or stochastic, and it can be known in advance or revealed during the tour.
A taxonomy of vehicle routing problems based on these two dimensions is proposed by [16]. Four types of vehicle routing problems are then distinguished :

Static and deterministic problems Input is known in advance and doesn't change over time. This is the most studied type of problem, but it generally doesn't fit with real-world applications, where some information cannot be known beforehand.

Static and stochastic problems Here, some information is a stochastic variable which is revealed gradually during the execution of the tour. However, the a priori planned routes cannot change during the execution of the tour except in some special cases. For example, if the considered stochastic variable is the customers request, or in other words, if customers may request a visit with a certain probability, the a priori planned route may change only to skip customers that do not require a visit. Several types of stochastic variables have been studied in the literature : stochastic travel times [17], where travel times between customers is a random variable, stochastic customers, where customers may request a visit with a certain probability [3]...

Dynamic and deterministic problems Some information is totally unknown beforehand and is revealed only during the execution of the tour. Vehicle tours are then changed in real time, during the execution of the tour according to revealed information.

Dynamic and stochastic problems This type of problem is a combination of the latter two types described above. Some information is a stochastic variable that can be used to build a priori tours taking into account possible future events, and routes are adapted in real time according to revealed information.

For more details on stochastic and dynamic vehicle routing problems, we refer the reader to the surveys of [16] and [1].

2.3 Stochastic/Dynamic pick-up & delivery problems

Most studies tackling pick-up & delivery problems consider the static case in which all information is known beforehand and does not change during time. However, some papers deal with the dynamic case where some information is only revealed during the tour and the a priori tour is adapted progressively in real time. A few of these works exploit stochastic information to anticipate future events, [18] present some of these papers. However, to the best of our knowledge, there is no work dealing with the One-commodity travelling salesman problem in a stochastic case.

The problem considered in this paper can be classified as a static and stochastic One-commodity travelling salesman problem. It is static because changes are not allowed during the execution of tour except for detours to recycling centres. Stochastic because we have probabilistic information through historical data about potential future detours. It is a One-commodity travelling salesman problem because a single-vehicle has to carry one commodity from a set of pick-up customers to a set of delivery customers.

3 Problem Formulation

The pick-up & delivery travelling salesman problem(1-PDTSP) can be defined on a complete graph $G = (V, E)$ as follows :

- $V = \{0, 1, ..., n\}$ is a set of $n + 1$ nodes representing the n customers ($n = n_d + n_p$, where n_d is the number of delivery customers and n_p the number of pick-up customers). Node 0 represents the depot ;
- $E = \{(i, j), i, j \in V, i \neq j\}$ is a set of edges representing connections between customers ;
- $C = \{c_{i,j}, (i, j) \in E\}$ represents the travel distance between customers i and j ($c_{i,j} = c_{j,i}, \forall (i, j) \in E$) ;
- $D = \{d_i, i \in V\}$ is a set of customers demands ($|d_i|$ is the number of bins to deliver to / pick up from customer i, $d_i < 0$ for delivery customers and > 0 for pick-up customers) ;

Given a vehicle with a known and finite maximum capacity Q, and assuming that :

- $x_{i,j}$ is a boolean variable such that:

$$\begin{aligned} x_{i,j} &= 1 && \text{if customer } j \text{ is visited immediately after customer } i; \\ x_{i,j} &= 0 && \text{otherwise.} \end{aligned}$$

- q_i the number of bins in the vehicle after his visit to customer i.

Our objective is to find a Hamiltonian cycle that minimizes the total travel distance, ie :

$$min \sum_{i=0}^{n} \sum_{j=0}^{n} x_{i,j} c_{i,j} \tag{1}$$

Subject to :

$$\sum_{j=0}^{n} x_{i,j} = 1 \qquad \forall i \in \{0, 1, ..., n\} \tag{2}$$

$$\sum_{i=0}^{n} x_{i,j} = 1 \qquad \forall j \in \{0, 1, ..., n\} \tag{3}$$

$$q_i + x_{i,j} d_j \leq Q \qquad \forall i, j \in \{0, 1, ..., n\} \tag{4}$$

$$q_i + x_{i,j} d_j \geq 0 \qquad \forall i, j \in \{0, 1, ..., n\} \tag{5}$$

Constraints (2) et (3) ensure that each customer is visited exactly once, while constraints (4) and (5) relate to vehicle capacity.

A picked up bin can be supplied to a delivery customer if necessary. However, if a bin is not totally clean and empty when it is picked up from a customer, it must be firstly routed to one of the R available recycling centres around the customer's location before it can be supplied again. We can consider the R available recycling centres as a "priority customer" that may requires a visit after each of the n_p pick-up customers. Then, we define p_i as the probability that the "priority customer" requires a visit immediately after customer i ($p_i = 0$ for all delivery customers). The problem can then be seen as a travelling salesman problem with stochastic customer requests.

4 Estimation-based algorithm

To tackle the 1-SPDTSP described above, we propose an estimation-based heuristic adapted from the approach presented in [2] for the probabilistic travelling salesman problem.
This approach is based on a local search method which starts from an initial feasible solution S, and tries to improve it by moving to S', a feasible neighbouring solution of S, such that $f(S') < f(S)$. The process is repeated until no improvement can be found.

```
$ImprovedSolution \leftarrow True$;
$S \leftarrow IntialSolution$;
while $ImprovedSolution$ do
    $N \leftarrow Neighborhood\ (S)$;
    $N \leftarrow RomoveUnfeasibleSolutions\ (N)$;
    for $S' \in N$ do
        if $f(S') < f(S)$ then
            $S \leftarrow S'$;
        end
    end
    $ImprovedSolution \leftarrow False$;
end
```

Algorithm 1: Local search principle

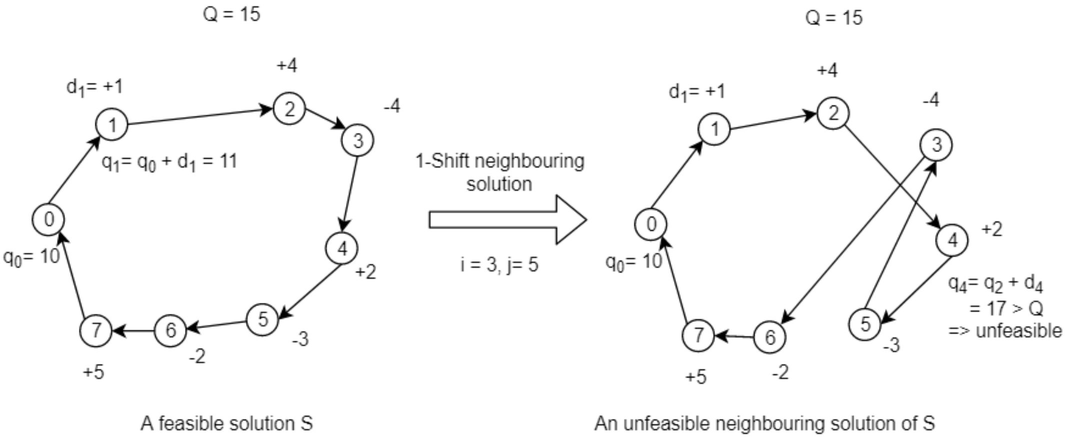
4.1 Neighbourhood structure

We use the 1-shift algorithm introduced in [3] to generate the neighbourhood of a given solution S. This method consists in changing the position of a customer in a tour from i to j. Customers which are at positions $i+1, i+2, ..., j$ of the tour are then shifted backwards (see figure 1).

4.2 Feasibility checking

For each generated solution, we unsure that capacity constraints described in section 3 are respected. A feasible solution is a tour in which the total number of bins loaded on the vehicle never exceeds the maximum capacity Q of the vehicle, and is never negative. Assuming that q_i is the number of bins in the vehicle after visiting customer i, figure 1 presents an example of feasible and infeasible solution.

Given a feasible solution S and an 1-shift neighbouring solution S' of S obtained by shifting a customer from position i to j. It can easily be shown that S' is feasible if and only if the partial tour from customer i to customer j is feasible. Indeed, to check to feasibility of a neighbouring solution, we only check the feasibility of the tour between positions i and j.



A feasible solution S

An unfeasible neighbouring solution of S

Fig. 1: 1-Shift algorithm

4.3 Objective function

In our case, the objective function f to minimize is the total travel distance of the vehicle. However, since we cannot know in advance the travel distance of an a priori solution S due to potential detours to recycling centres (see figure 2), we use the following unbiased estimator of $f(S)$ as a criterion to move from a solution to another one :

$$\hat{f}_M(S) = \frac{1}{M}\sum_{r=1}^{M} f(S,\omega_r)$$

This estimator was proposed by [2] for the probabilistic travelling salesman problem. The idea is to estimate the quality of an a priori solution S from a set of M simulations of possible a posteriori solutions. An a posteriori solution is obtained by associating a binary vector ω with the a priori solution such that $\omega[i] = 1$ if a detour to a recycling centre is required immediately after visiting customer i, 0 otherwise (see vector ω in figure 2).
Thus, given an a priori solution S (that does not include recycling detours) :

1. M possible a posteriori solutions (including potential detours) are generated by associating M vectors ω with the a priori solution S;
2. For each generated a posteriori solution, $f(S,\omega_i)$, the travel distance of the a posteriori solution given by ω_i is calculated. $f(S,\omega_i) = f(S) + TDL - \sum_{i=1}^{n}\sum_{j=1}^{n} \omega_i x_{i,j} c_{i,j}$, where :
   - $f(S)$ is the travel distance of the a priori solution S (without detours) ;
   - TDL is the Total Detour Length of the a posteriori solution (see example in figure 2).
3. $\hat{f}_M(S) = \frac{1}{M}\sum_{r=1}^{M} f(S,\omega_r)$ is calculated and considered as an estimator of $f(S)$.

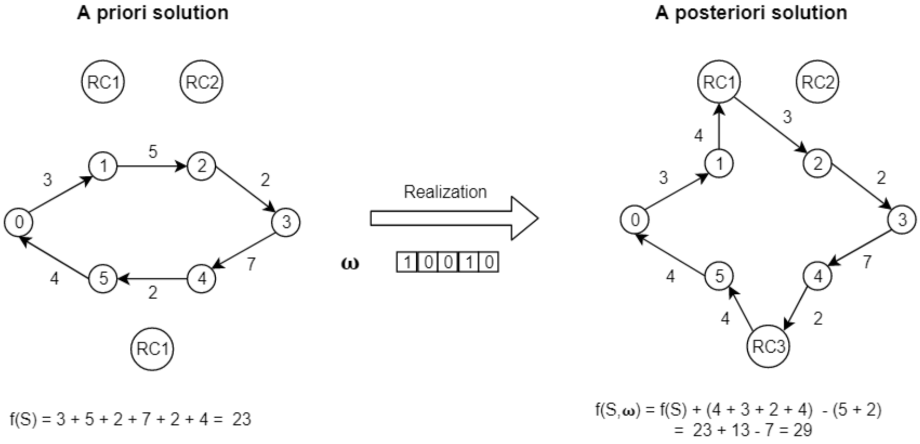


Fig. 2: A priori solution VS a posteriori solution

Note that ω is generated according to the set $P = \{p_i, i \in V\}$ of probabilities that recycling detour is required after visiting customer i. Therefore, $\omega[i] = 0$ for all delivery customers because recycling detour may occur only when picking up bins.

4.4 Recycling centre choice

Since we consider R available recycling centres in our problem, each time a detour to recycling centre is required, we must choose among the R possibilities we have. Therefore, we calculate the travel distance caused by the detour to each of the R available recycling centres to choose the one that minimizes the detour length (see figure 3).

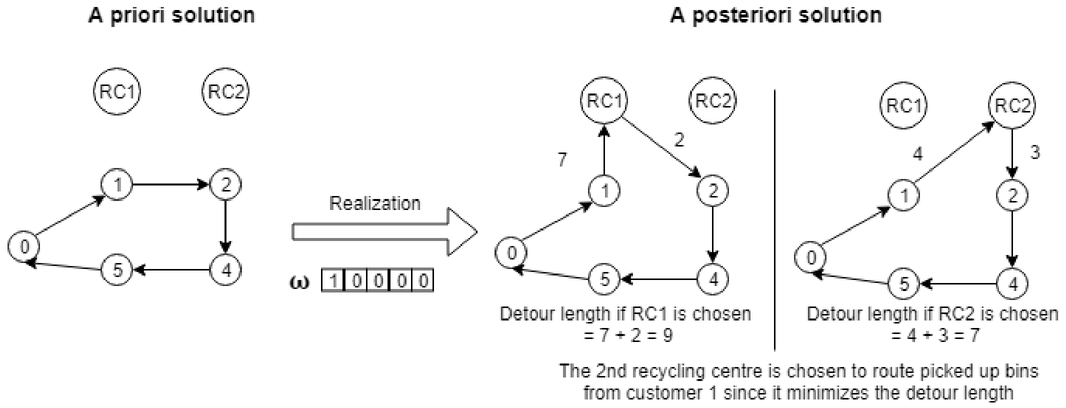

A priori solution
A posteriori solution
RC1
RC2
0
1
2
4
5
Realization
ω
1 0 0 0 0
7
2
4
3
Detour length if RC1 is chosen = 7 + 2 = 9
Detour length if RC2 is chosen = 4 + 3 = 7
The 2nd recycling centre is chosen to route picked up bins from customer 1 since it minimizes the detour length

Fig. 3: Recycling centre choice

5 Computational results

The algorithm was implemented in Java, and executed on AMD A10-7700K Radeon R7, 3.40 GHz With 8 GB RAM.

We tested the performance of our algorithm on the Euclidian PDTSP instances generated by [19]. The number of customers in these instances vary between 25 and 200. The first four customers of each instance have been chosen to be the recycling centres, the remaining nodes are assumed to be the customers. For each customer i, we determined whether a recycling detour is required after visiting i or not (we determined an "effective scenario" for each instance). The boolean variables were generated according to a fixed probability P. We generated scenarios for $P \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$. Then, the a priori solutions found by our algorithm were evaluated according to the fixed scenarios, with different values for the parameter M, the number of simulated a posteriori solutions (see section 4.3). For $M = 0$, our algorithm doesn't simulate a posteriori solutions. It is then equivalent to a classic local search which doesn't take into account stochastic information.

Table 1 shows the average solution cost obtained by our Estimation-Based Local Search for the instances described above (we fixed the neighbourhood size to 200).

First, we observe that, for each class of instances, solution costs increase as the parameter p increases. This is due to the fact that a higher probability p involves a greater risk of requiring detours to recycling centres and thus, a greater risk of increasing the solution cost. However, we can see that this increase is smaller as the number of generated a posteriori solutions (M) is greater.

Figure 4 shows the percentage of travel distance due to detours when $p = \{0.1, 0.3, 0.5, 0.7, 0.9\}$, and for $M = \{0, 25, 50, 100\}$. The results show the effectiveness of our approach in minimizing the detours impact on the solution cost, especially when $p \geq 0,5$. Indeed, the objective of our estimation-based local search is to anticipate possible detours during the tour and take them into account when generating a priori solutions. Therefore, the more detours may occur during a tour, the more interesting our approach is. In other words, and as we can observe in table 1, our estimation-based heuristic always obtains the best solutions in comparison with the classic local search (the one with $M = 0$) when $p > 0,1$. Moreover, the results are generally better when $M = 100$. Thus, a greater number of a posteriori simulations gives generally a more accurate evaluation of an a priori solution.

6 Conclusion and perspectives

We presented in this a paper an Estimation-based local search to tackle a stochastic one-commodity pick-up & delivery travelling salesman problem. The objective of our approach is to build efficient vehicle routes that minimize loss of quality due to potential changes during the tour. We tested our algorithm on the Euclidian PDTSP instances proposed in [19]. We adapted the instances to fit our constraints and collected the results with different parameter values. The experiments show the effectiveness of our algorithm, especially when dealing with large instances, and when detours

Table 1: Estimation-based local search solutions for the Euclidian PDTSP instances

		Number of sample solutions M			
Probability	Number of customers	0	25	50	100
$p = 0.1$	25	**544,26**	555.9	551,5	565,4
	50	**843,33**	895	893.1	860.6
	75	1121,1	1119	1138,66	**1111,5**
	100	**1414,56**	1432,5	1494	1457,5
	150	2007,07	2017	2013	**1993,3**
	200	2603,73	2601.1	2496	**2489,16**
$p = 0.3$	25	618,5	596	**574,26**	577,3
	50	1020,03	1018,2	1051,3	**1001,1**
	75	1446,9	1474,4	1398,12	**1392,8**
	100	1874,26	1862,2	**1813.7**	1841,1
	150	2673,83	2641,14	2594	**2569,4**
	200	3592,26	3617,14	3504,2	**3495,6**
$p = 0.5$	25	706,66	688,3	681.13	**680,6**
	50	1281,33	1210,15	1187	**1161,7**
	75	1848,86	1817,21	1832,2	**1804,4**
	100	2233,96	2214,1	2157	**2149,4**
	150	3416,86	3411,14	**3378,9**	3386.6
	200	4445,53	4431,37	4376,1	**4348,4**
$p = 0.7$	25	782,06	771	**734,3**	746,2
	50	1480,83	1535,2	1457,5	**1447,3**
	75	2172	2267,3	2169	**2125,2**
	100	2666,5	2517,3	2500,1	**2491,7**
	150	4047,16	4006,7	4011,6	**3992,8**
	200	5490,2	5397,3	**5325,8**	5332,3
$p = 0.9$	25	867,8	804,4	**786,4**	799,1
	50	1682,06	1633,9	**1526,1**	1589,1
	75	2472	2480,4	2366,5	**2306,2**
	100	3108,8	3116,5	2915,2	**2903**
	150	4741,8	4886,3	4605,36	**4552,52**
	200	6457,8	6384,8	6376,6	**6301,1**

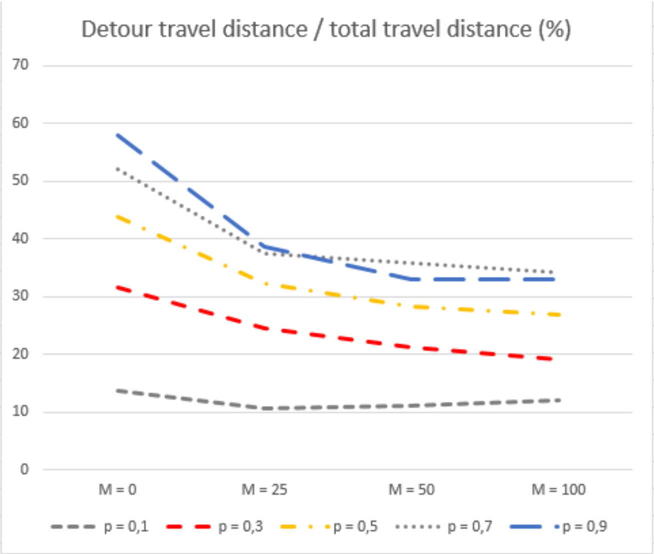


Fig. 4: Recycling detour's impact on total travel distance

are more likely to occur.
In this paper, we proposed a static and stochastic approach to tackle our vehicle routing problem. To improve further the obtained results, future works will be devoted to the development of a dynamic and stochastic approach which can exploit stochastic information to build efficient routes that can dynamically change to fit potential unexpected events during the vehicle tour.

References

1. RITZINGER, Ulrike, PUCHINGER, Jakob, et HARTL, Richard F. A survey on dynamic and stochastic vehicle routing problems. International Journal of Production Research, 2016, vol. 54, no 1, p. 215-231.
2. BIRATTARI, Mauro, BALAPRAKASH, Prasanna, STTZLE, Thomas, et al. Estimation-based local search for stochastic combinatorial optimization using delta evaluations: a case study on the probabilistic traveling salesman problem. INFORMS Journal on Computing, 2008, vol. 20, no 4, p. 644-658.
3. BERTSIMAS, Dimitris. Probabilistic combinatorial optimization problems. 1988. Thse de doctorat. Massachusetts Institute of Technology.
4. PSARAFTIS, Harilaos N. A dynamic programming solution to the single vehicle many-to-many immediate request dial-a-ride problem. Transportation Science, 1980, vol. 14, no 2, p. 130-154.
5. CORDEAU, Jean-Franois. A branch-and-cut algorithm for the dial-a-ride problem. Operations Research, 2006, vol. 54, no 3, p. 573-586.
6. TOTH, Paolo et VIGO, Daniele. Fast local search algorithms for the handicapped persons transportation problem. In : Meta-Heuristics. Springer, Boston, MA, 1996. p. 677-690.
7. CORDEAU, Jean-Franois et LAPORTE, Gilbert. A tabu search heuristic for the static multi-vehicle dial-a-ride problem. Transportation Research Part B: Methodological, 2003, vol. 37, no 6, p. 579-594.
8. GRIBKOVSKAIA, Irina, HALSKAU SR, yvind, LAPORTE, Gilbert, et al. General solutions to the single vehicle routing problem with pickups and deliveries. European Journal of Operational Research, 2007, vol. 180, no 2, p. 568-584.
9. HOFF, Arild et LKKETANGEN, Arne. Creating lasso-solutions for the traveling salesman problem with pickup and delivery by tabu search. Central European Journal of Operations Research, 2006, vol. 14, no 2, p. 125-140.
10. CHEN, Jeng-Fung et WU, Tai-Hsi. Vehicle routing problem with simultaneous deliveries and pickups. Journal of the Operational Research Society, 2006, vol. 57, no 5, p. 579-587. and pickups
11. HERNNDEZ-PREZ, Hiplito et SALAZAR-GONZLEZ, Juan-Jos. A branch-and-cut algorithm for a traveling salesman problem with pickup and delivery. Discrete Applied Mathematics, 2004, vol. 145, no 1, p. 126-139.
12. HERNNDEZ-PREZ, Hiplito et SALAZAR-GONZLEZ, Juan-Jos. Heuristics for the one-commodity pickup-and-delivery traveling salesman problem. Transportation Science, 2004, vol. 38, no 2, p. 245-255.
13. HERNNDEZ-PREZ, H. Traveling salesman problems with pickups and deliveries. Disertation, University of La Laguna, Spain, 2004.
14. HERNNDEZ-PREZ, Hiplito, RODRGUEZ-MARTN, Inmaculada, et SALAZAR-GONZLEZ, Juan Jos. A hybrid GRASP/VND heuristic for the one-commodity pickup-and-delivery traveling salesman problem. Computers & Operations Research, 2009, vol. 36, no 5, p. 1639-1645.
15. BERBEGLIA, Gerardo, CORDEAU, Jean-Franois, GRIBKOVSKAIA, Irina, et al. Static pickup and delivery problems: a classification scheme and survey. Top, 2007, vol. 15, no 1, p. 1-31.
16. PILLAC, Victor, GENDREAU, Michel, GURET, Christelle, et al. A review of dynamic vehicle routing problems. European Journal of Operational Research, 2013, vol. 225, no 1, p. 1-11.
17. KENYON, Astrid S. et MORTON, David P. Stochastic vehicle routing with random travel times. Transportation Science, 2003, vol. 37, no 1, p. 69-82.
18. BERBEGLIA, Gerardo, CORDEAU, Jean-Franois, et LAPORTE, Gilbert. Dynamic pickup and delivery problems. European journal of operational research, 2010, vol. 202, no 1, p. 8-15.
19. GENDREAU, Michel, LAPORTE, Gilbert, et VIGO, Daniele. Heuristics for the traveling salesman problem with pickup and delivery. Computers & Operations Research, 1999, vol. 26, no 7, p. 699-714.

APM-MOEA : An asynchronous parallel model for multi-objective evolutionary algorithms

Florian Mazière[1,2], Pierre Delisle[2], Caroline Gagné[1], and Michaël Krajecki[2]

[1] Département d'informatique et de mathématique
Université du Québec à Chicoutimi, Chicoutimi, Canada, G7H 2B1
{florian.maziere1,caroline.gagne}@uqac.ca
[2] CReSTIC, Université de Reims Champagne-Ardenne
Reims, France, 51687
{pierre.delisle,michael.krajecki}@univ-reims.fr

Abstract. Despite their proven effectiveness, current parallel models for Multi-Objective Evolutionary Algorithms (MOEAs) often struggle in reaching both a good Pareto front approximation and a high level of performance on supercomputer architectures. In this work, we propose APM-MOEA, an Asynchronous Parallel Model for MOEAs. It is based on an island model with objective space division. The main features of the proposed model are a global view for the organizer to achieve better distribution of solutions, an asynchronous communication scheme to reduce model overhead, control islands to improve diversity and a local search procedure to improve the quality of solutions. Extensive experiments have been conducted using the GISMOO algorithm to compare APM-MOEA with state-of-the-art island-based models in the resolution of the multi-objective travelling salesman problem (MOTSP). The experimental results show that, according to four multi-objectives metrics, our approach outperforms all the implemented parallel models in terms of convergence, diversity of solutions and scalability.

1 Introduction

Many academic and industrial optimization problems are multi-objective and have been of particular interest to researchers in recent years. These problems usually do not have a single optimal solution but a set of best trade-off solutions which form the so-called Pareto front in the objective space. In order to approximate this Pareto front, multi-objective evolutionary algorithms (MOEAs) have been largely investigated in the fields of continuous and combinatorial optimization [1]. As they often require a high amount of computing resources to explore large portions of the search space and handle complex real-life constraints, they could greatly benefit from today's high-performance computing architectures. Thus, parallel multi-objective evolutionary algorithms (pMOEAs) have been proposed in the literature to reduce computation times and improve the quality of the obtained Pareto fronts [2]. Among the pMOEAs paradigms, master-slave and distributed models are certainly the most studied [3]. Master-slave models aim to reduce the execution time of MOEAs by parallelizing their main operations such as crossover, mutation and evaluation functions. Island models, also called distributed models, mainly aim to reach the Pareto front more effectively and improve the quality of obtained solutions. In such a model, the population is divided into islands which run an independent MOEA and carry out information exchanges to enhance the exploration capabilities of the algorithms. Although significant progress has been made in recent years in the design and improvement of parallel models for evolutionary algorithms, most of these models have limited scalability and ability to solve various problems. In fact, solving multi-objective combinatorial optimization problems efficiently on a large number of processors remains a challenge today.

This paper aims to propose a new island model, namely APM-MOEA (Asynchronous Parallel Model for Multi-Objective Evolutionary Algorithms), which is based on objective space division and inspired by the works of Streichert *et al.* [4]. The main features of the proposed model are the following :

- Each island is assigned to a specific part of the objective space according to a principle of constrained dominance ;

- An organizer has a global view of the current search via a global archive ;
- A specific clustering algorithm is used to divide the objective space and achieve a better distribution of solutions ;
- Asynchronous cooperation between islands, especially for the exchange of local archives with the organizer, to limit model overheads ;
- Control islands to guide the exploration of the search space and improve diversity ;
- A periodic use of a specific local search procedure to improve convergence.

In this work, APM-MOEA is used to parallelize the GISMOO algorithm [5], which has proven to be efficient in solving classical multi-objective optimization problems. In order to evaluate the performance of the proposed model, other state-of-the-art island models are also adapted to GISMOO and a comparative study is conducted on the resolution of the Multi-Objective Travelling Salesman problem (MOTSP) [6].

The remainder of this paper is organized as follows : the next section provides an overview of the main existing distributed models for MOEAs. In Section 3, the main features of the APM-MOEA model are outlined. Experimental conditions, results and analysis are exposed in Section 4. Finally, the last section contains some conclusions and new research directions.

2 Related works on Parallel Multi-Objective Evolutionary Algorithms

During the last few years, several models have been designed to parallelize MOEAs [3]. While master-slave models focus on reducing execution time, islands models aim to improve solution quality and diversity. In such a model, the global population is partitioned into several islands and a single MOEA runs on each of them, usually on a separate processor. Based on this concept, the PMOMA algorithm [7] uses a ring topology to exchange the best non-dominated solutions between subpopulations and the Multi-Front algorithm [8] exchanges the first fronts obtained by a Pareto ranking, highlighting the difficulty of determining the best number of fronts to exchange which depends on the shape of the Pareto front. More recently, Sanhueza *et al.* [9] proposed PasMoQAP, an asynchronous island model to solve the multi-objective quadratic assignment problem. Every five generations, each island shares two promising individuals with other islands and integrates the received individuals in an elitist way. Promising results are obtained with up to 11 islands and compared to a self-made implementation of an island model of NSGA-II [10].

The cooperative coevolution strategy was also used to distribute the computation among the islands. In such an approach, each subpopulation is focused on the optimization of a subset of the problem variables. Dorronsoro *et al.* [11] have adapted the original cooperative coevolution to the multi-objective context and proposed CCMOEA. With three state-of-the-art multi-objective algorithms, the parallel model allows to obtain better solutions using 4 or 8 islands on a real-world combinatorial optimization problem. Atashpendar*et al.* [12] hybridized cooperative evolution with a particle swarm optimization algorithm in order to efficiently tackle well-known continuous problems.

In the context of multi-objective optimization, the main goal is usually not to seek a single optimum solution but rather a set of Pareto optimal solutions. This provides the opportunity to design a parallel model in which each processor search for different solutions and cooperate to approximate the whole Pareto front. Thus, some parallel models use a *divide-and-conquer* strategy to find non-dominated solutions and most of them are based on objective space division. The first model which tries to specialize the processors to specific parts of the objective space is DRMOGA [13]. At regular intervals, a global population is formed from the subpopulations, sorted according to one objective function and redistributed on the subpopulations. De Toro Negro *et al.* [14] use a similar approach in PSFGA and propose two mechanisms to maintain diversity within the whole population. Deb *et al.* [15] give different search directions to each island using the guided dominance principle. A small number of non-dominated solutions are periodically exchanged to enhance diversity. Results show that without prior knowledge about the shape of the Pareto front, it may be difficult to define good search directions.

In the previous models, islands cooperate to approximate the whole Pareto front and some mechanisms are used to limit overlaps. To completely avoid them, two other models explicitly divide the objective space. The first model is the cone separation model [16]. It uses a geometrical approach to subdivide the objective space which is effective if the Pareto optimal front is continuous and convex. The second model has been proposed by Streichert *et al.* [4] and applies a subdivision scheme based on a k-means clustering algorithm. All subpopulations are periodically gathered, clustered and redistributed onto the available islands. The main drawback of the approach is that the gathering of all subpopulations produces a high communication overhead, in particular with a high number of processors [17].

Despite their proven effectiveness on continuous problems, these models often struggle in reaching both a good Pareto front approximation and high scalability [3]. Moreover, according to our knowledge, they have not all been experimented on combinatorial optimization problems. Also, current state-of-the-art parallel models are generally experimented with 2 or 4 processors and show reduced effectiveness on larger numbers of processors [18]. In this paper, we propose APM-MOEA, a new pMOEA which handles multi-objective combinatorial optimization problems using larger numbers of processors.

3 The APM-MOEA model

APM-MOEA is an island-based model which uses a clustering algorithm to divide the objective space and assign specific parts of it to each island. It uses three different kinds of islands : one organizer (or master), a number of normal islands and two control islands. Fig. 1 illustrates the APM-MOEA model using 6 islands and highlights the communications from the organizer to the other islands. An example of partitioning for this model is given in Fig. 2 with the corresponding clusters and centroids.

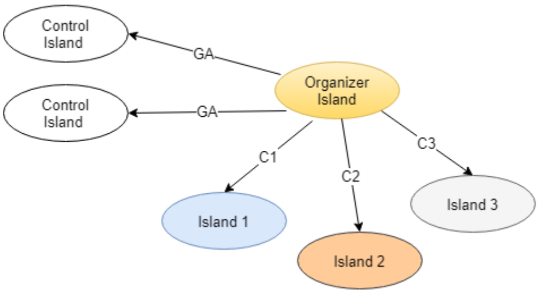


Fig. 1: Scheme of APM-MOEA using 6 islands (**GA** : Global archive, **CX** : Cluster X)

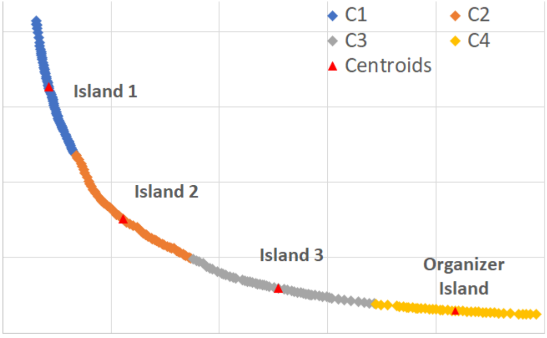


Fig. 2: Example of clustering for APM-MOEA using 6 islands, 4 clusters for the 3 normal islands and 1 for the organizer

The overall behavior of the proposed model is defined as follows. After the initialization of the populations and the archives used to store the non-dominated solutions, an iterative process is repeated until a stopping criterion is reached which generally is a maximum number of generations or a maximum execution time. First, the populations evolve through genetic operations during a number of generations where each island is focused on a specific region. During the genetic operations, the local archives, one for each island, are continually updated with the new generated solutions. A local search procedure is then applied on each population to improve the quality of solutions. The next steps are managed by the organizer to redefine the regions of the objective space for each island. Although it follows the outlines of the model of Streichert *et al.*[4], APM-MOEA implements significant changes which are described in the following sections.

3.1 Global search view of the organizer

The organizer island is responsible for handling the clustering and has a global view of the current search. It maintains a global archive that is updated with local archives from all other islands. At each migration point, the global archive is partitioned by a clustering algorithm, then clusters and centroids are distributed to their respective islands. The main loop of the organizer is illustrated in Fig. 3. The other islands have a similar outline, but they only send their local archives when a new migration point is reached and check for a new cluster at regular intervals. It is important to note that the solutions from each cluster are not necessarily integrated in the population since an elitist replacement is done to keep only the best solutions.

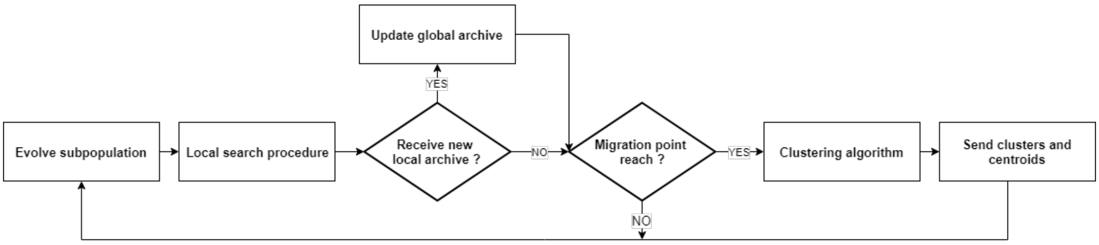


Fig. 3: Main loop of the organizer island

The original model of Streichert *et al.* uses a classic k-means and has the drawback of relying heavily on randomly selected initial centroids. Other methods have been proposed to improve these initial points and thus improve the algorithm efficiency. Among these, Yedla *et al.* [19] introduced a new clustering algorithm which includes a method for finding better initial points and a more efficient approach to assign the data points to the corresponding clusters. Experiments have shown that their algorithm has more accuracy and is faster than the original k-means clustering algorithm. The main features of this clustering algorithm are integrated into APM-MOEA and the origin point is replaced by the ideal vector to take into account the characteristics of the multi-objective problems.

In order to limit each island to a specific region of the objective space during the evolution of populations, constraints are added to the problem. Thus, a solution S is feasible if and only if the nearest centroid from S is the centroid assigned to the considered island. In other cases, the solution S is marked as invalid and is not favored in the evolutionary operations. For example, in the context of the MOTSP, the constraint violation can be the Euclidean distance between the objective vector and the centroid attributed to the island. The constrained domination principle introduced by Deb *et al.* [10] modifies the definition of domination and it is used to limit the islands to their own region. More precisely, it is used to compare two solutions in the binary tournament of the MOEA, to select the best offspring in the genetic phase and to sort the population in the replacement step. It is important to mention that the original Pareto dominance is still used to update the local archives.

3.2 Asynchronous communications

As it has been pointed out by Jaimes *et al.* [17], the main drawback of the Streichert model is the communication overhead caused by the exchange of populations. To address this issue and handle the irregularity of local search computations, the APM-MOEA model uses asynchronous communications to avoid using a global synchronization barrier. As shown in Fig. 3, the organizer island does not have to wait for all the local archives to perform the clustering. At regular intervals, it simply checks if new local archives have been sent. If this is the case, the global archive is updated with the new solutions. All send operations are non-blocking and therefore do not need to wait for the corresponding receive operations. The communication overhead for the other islands is also limited as they only have to periodically check for new clusters and send their local archive.

3.3 Control islands

Two additional islands, named control islands, are included in APM-MOEA in order to find more diversified solutions and explore the objective space more efficiently. Unlike the other islands, they have no constraint about the search area. The first control island includes the local search procedure in its process while the second generates new solutions only with evolutionary operations.

These islands allow to explore the whole objective space with a global view of the current search. The control islands communicate exclusively and asynchronously with the organizer island which provides new solutions contained in its global archive (cf. Fig. 1). Like the normal islands, they periodically send their local archive to the organizer.

3.4 Local Search

The performance of multi-objective algorithms can often be improved through the inclusion of some form of local search. In this sense, hybrid algorithms specifically tailored to combinatorial multi-objective problems have been developed [20] [9] [21].

In APM-MOEA, a non-iterative local search is applied to improve each solution of the population at regular intervals. Partial neighborhood exploration [22] is favored rather than exhaustive exploration to reduce computation time. Depending on problem characteristics, two strategies can be used : a first improving strategy and a candidate list strategy [23]. The former tries to generate all the neighbors and stops when a solution which dominates the original solution is found. A maximum number of tries can be set to limit computation overhead. The latter generates a subset of all the neighbors associated to a candidate list by keeping all the neighbors that dominate the current solution in an additional archive. At the end of the search process, a single non-dominated solution from the archive is randomly selected to replace the current solution in the population.

4 Experimental Results

In a first step, experiments aim to compare the performance of APM-MOEA with three state-of-the-art island-based models which use a divide-and-conquer approach : the cone separation model [16] (`Cones`), the DRMOGA model [13] (`DRMOGA`) and the original model of Streichert *et al.* [4] (`Original`). The proposed model have been tested with control islands (`APM-C`) and without them (`APM`). The quality of obtained solutions and the scalability are analyzed through four metrics that are presented later. For a fair comparison, local search is incorporated in the same way for all models. After focusing on the scalability of APM-MOEA in a second step, the contribution of the control islands is finally emphasized. Beforehand, we present the experimental design used in this study.

4.1 Multi-objective Travelling Salesman Problem

A well-known benchmark in multi-objective combinatorial optimization is used for these experiments : the Multi-Objective Travelling Salesman Problem [6]. In its mono-objective version (TSP), given a number of cities and a distance between each pair of cities, the travelling salesman problem is to find the shortest way of visiting all the cities and returning to the starting point. In its multi-objective version, there are different kinds of costs between each pair of cities, each one being associated to a specific cost matrix.

For the experiments, six combinations of single-objective TSP instances from TSPLIB [24] are tested ranging from 200 to 1000 cities. These instances are referred as kroAB200, kroAB300, kroAB400, kroAB500, kroAB750 and kroAB1000. For example, kroAB200 is a bi-objective instance with 200 cities and represents the combination of kroA200 (the 1st objective) and kroB200 (the 2nd objective). As large instances of MOTSP are experimented, the optimal Pareto fronts are not known. Therefore, state-of-the-art solution sets provided by Lust and Teghem [21] are used as a reference set.

4.2 The GISMOO algorithm

The GISMOO algorithm [5] was chosen to evolve the subpopulations and to illustrate the performance of APM-MOEA. During the last years, it has proven to be efficient to solve a variety of continuous [25] [5] or combinatorial [26] [27] problems. More specifically, its immune phase allows to obtain well-diversified solutions which seems to be a significant benefit for a divide-and-conquer model. Following the authors guidelines, all GISMOO experimentations were carried out with the following parameters : population size, mutation probability and crossover probability are respectively set to 100, 0.06 and 1.0.

4.3 Experimental conditions

All parallel models are implemented in the C++ language using the MPI library and the *mpiicpc* compiler. They are executed on one node of the ROMEO supercomputer [28] which allows the use of 2, 4, 8 and 16 processor cores for these experiments. For each parallel model, an unique processor core is associated to one island. All test instances were solved 5 times for each parallel model and the average of the metric values are presented.

In regard to the evolutionnary phase, the order crossover (OX) is used to perform the recombinations and the inversion mutation is used as the mutation operator. The 2-opt neighborhood Pareto local search with the speedup technique proposed by Lust and Teghem [21] is implemented in each model. The same maximum execution time is also set as the stopping criteria for each test instances and each model. More precisely, the computational times for the instances kroAB200, kroAB300, kroAB400, kroAB500, kroAB750, kroAB1000 are set to 200, 300, 600, 1000, 2500 and 5000 seconds respectively.

4.4 Performance metrics

In the multi-objective context, it is difficult to assess both convergence toward the Pareto front and diversity of the obtained solutions. However, in the last decades, many metrics have been proposed to compare the performance of different MOEAs [29]. Four metrics are used in these experiments :

– The generational distance GD [30] measures the average distance from obtained solutions to the reference set solutions. It allows the evaluation of the convergence of a solution set toward the Pareto front. Lower values of this metric are associated to better convergence ;
– The hypervolume metric H [31] measures the size of the portion of the objective space that is dominated by a given solution set. It allows the estimation of both the convergence and the diversity of the solutions. Higher values of hypervolume are preferable ;
– The coverage of two sets $\mathscr{C}$ compares the convergence of two given solution sets using Pareto dominance. More specifically, $\mathscr{C}(A, B)$ measures the proportion of solutions in set B which are dominated by solutions in set A ;
– The minimal spacing (ms) [32] evaluates the spread of solutions contained in a set. It addresses the limitations of classical spacing metrics by computing the distance from a solution to its nearest neighbor which has not already been considered. The lower the value is, the better is the diversity.

4.5 Comparison with others parallel models

We first compare the quality of the non-dominated solutions obtained by each model. Table 1 presents the mean deviations to the best values of GD obtained by all models using 2, 4, 8 and 16 islands. For each configuration, the best result is marked in bold characters. According to this metric, the cone separation model finds better solutions than other models using only 2 processor cores. With more islands, the two APM-MOEA variants outperforms the compared models, especially with 16 islands where `APM-C` obtains a mean value of 0.01 for the GD metric while the second best (`Original`) obtains only 0.42. Furthermore, for the APM-MOEA model, the GD metric significantly decreases with the number of islands, showing its ability to improve the quality of its solutions when the number of islands increases. On the contrary, the `Cones` model do not find better solutions with a large number of processor cores.

	DRMOGA	Cones	Original	APM	APM-C
2 isl.	2.27	**1.69**	2.28	2.12	N/D
4 isl.	1.20	1.20	1.32	**1.00**	1.62
8 isl.	0.80	0.79	0.61	0.43	**0.40**
16 isl.	0.53	0.92	0.42	0.13	**0.01**

Table 1: Means deviations to GD

	DRMOGA	Cones	Original	APM	APM-C
2 isl.	0.863	0.806	0.820	**0.897**	N/D
4 isl.	0.888	0.886	0.892	**0.909**	0.904
8 isl.	0.898	0.909	0.912	0.918	**0.919**
16 isl.	0.900	0.910	0.914	0.921	**0.923**

Table 2: Means of hypervolume metric H

To confirm the results obtained with the previous metric, we computed the hypervolume metric for all models. In each configuration, `APM` and `APM-C` obtain a greater value of H than state-of-the-art models. Once again, the metric values are significantly better with 16 islands. These results show that the portion of the objective space which is dominated by the solutions provided by APM-MOEA is generally greater than other models, indicating a better convergence to the Pareto front and a good diversity within the obtained solutions.

The results of these first two metrics seem to indicate that APM-MOEA manages to find solutions closer to the Pareto front than other models, especially for configurations with a large number of islands. To confirm this assumption, we directly compared the models with each other on the coverage $\mathscr{C}$ metric. We computed the values of $\mathscr{C}$ for `APM-C` compared to `Cones` and `Original`. Fig. 4 and Fig. 5 outline, for each test instance, the average $\mathscr{C}$ values obtained using 8 and 16 islands respectively.

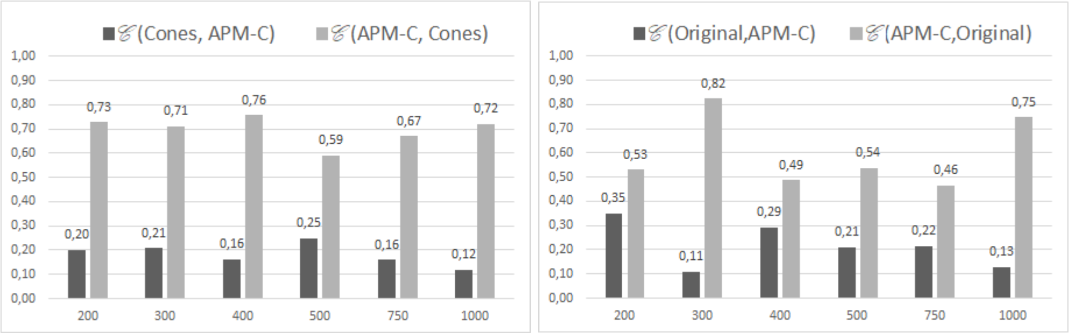


Fig. 4: Average coverage $\mathscr{C}$ of `Cones`, `Original` and `APM-C` using 8 islands

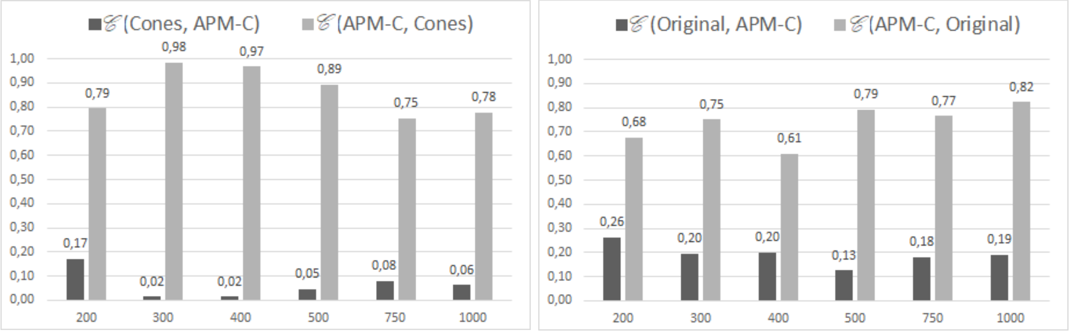


Fig. 5: Average coverage $\mathscr{C}$ of `Cones`, `Original` and `APM-C` using 16 islands

The results obtained with this metric significantly favors APM-MOEA. In fact, the average values obtained by $\mathscr{C}$(`APM-C`, `Cones`) and $\mathscr{C}$(`APM-C`, `Original`) are greater than the values of $\mathscr{C}$(`Cones`,`APM-C`) and $\mathscr{C}$(`Original`,`APM-C`) for all test instances. This trend is further confirmed when using 16 islands. Average values of $\mathscr{C}$(`APM-C`,`Cones`) vary between 0.75 and 0.98, implying that most solutions obtained by the `Cones` model are dominated by the solutions of the proposed model. On the contrary, the values $\mathscr{C}$(`Cones`,`APM-C`) are near to 0, showing that `APM-C` finds solutions that are not dominated by the solutions of `Cones`. The comparison with the `Original` model shows a similar tendency.

With the previous metrics, we have shown that APM-MOEA generally finds solutions closest to the Pareto front than state-of-the-art models. We now analyze the diversity of the obtained sets by computing the minimal spacing metric. Table 3 summarizes the average minimal spacing metric obtained by each model for MOTSP instances. With any number of islands, our model provides a better spread of solutions than any other models as shown by the low ms values obtained. Moreover, its diversity increases with the number of islands. In particular, by using an enhanced clustering algorithm to distribute the computations, APM-MOEA provides well-distributed solutions.

	DRMOGA	Cones	Original	APM	APM-C
2 isl.	0.00388	0.00160	0.00135	**0.00064**	N/D
4 isl.	0.00237	0.00061	0.00046	**0.00039**	0.00050
8 isl.	0.00167	0.00033	0.00034	0.00023	**0.00022**
16 isl.	0.00145	0.00030	0.00032	0.00019	**0.00014**

Table 3: Means of minimal spacing metric ms for each parallel model

Fig. 6 illustrates overall results by providing examples of solution sets obtained by the `Cones`, `Original` and `APM-C` models on a typical run solving a bi-objective MOTSP instance. The reader may note that APM-C obtains solution sets that are better distributed and that cover larger regions of the objective space, confirming the values obtained by the minimal spacing and the hypervolume metrics. These examples also highlight the good convergence of APM-MOEA which provides lower values for each objective function.

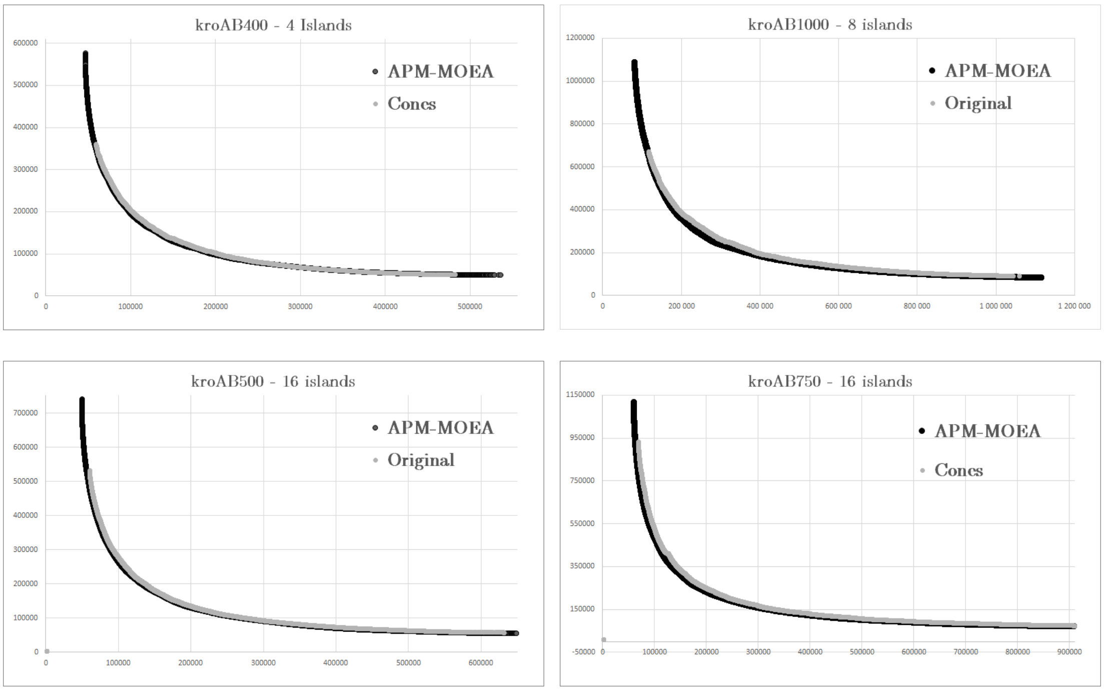


Fig. 6: Example of solutions set obtained by `Cones`, `Original` and `APM-C`

4.6 Scalability

Previous experiments have shown that increasing the number of islands also improve the quality of the solutions produced by APM-MOEA. In order to provide more insight about its scalability, we analyzed the evolution of the global archive through time with different numbers of islands. To enable this study, the global archive has been periodically saved by the organizer in an external file and the metric values were calculated afterwards to limit the computation overhead. Fig. 7 shows the evolution of convergence GD using 4, 8 and 16 islands for kroAB400 and kroAB750 instances. For each of the two test instances, a large number of islands allowed reaching a better convergence toward the Pareto front in a much faster way. In fact, the 16 islands configuration takes significantly less time than the two other ones to achieve a specific GD value. Moreover, the largest gap is observed between 4 and 8 islands. In the same way, Fig. 7 shows the evolution of the hypervolume metric H using 4, 8 and 16 islands for the same test instances. Once again, H values are higher and reached more quickly, showing a better distribution of the obtained solutions and a larger coverage of the objective space. Large gaps are also observed between 4 and 8 islands. Finally, results on hypervolume and minimal spacing metrics showed the ability of APM-MOEA to distribute the islands well on the objective space.

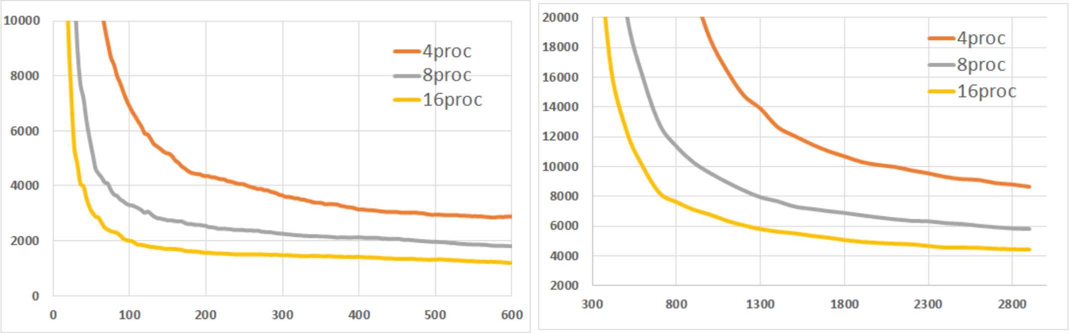


Fig. 7: Evolution of the GD metric of APM-C for kroAB400 (left) and kroAB750 (right)

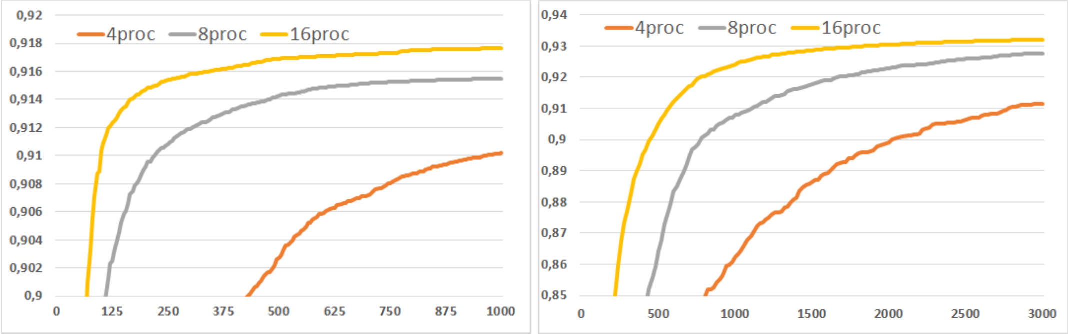


Fig. 8: Evolution of the H metric of APM-C for kroAB400 (left) and kroAB750 (right)

4.7 Contribution of the control islands

The last step of this study is to analyze the contribution of the control islands to the efficiency of APM-MOEA by comparing the two APM-MOEA variants through three of the multi-objective metrics previously presented.

Table 4 first exposes the average GD value obtained by `APM-C` and `APM` for each test instance. It shows that using control islands with a small number of islands is inefficient. In fact, `APM` obtains the lowest values of GD for all test instances, showing a better convergence. Using more islands, the APM-MOEA model with control islands is able to find solutions closer to the Pareto front for most test cases. since it obtains the best values of GD for 9 of the 12 cases. The reader may note that the best improvements are obtained with the largest test instances (kroAB750 and kroAB1000). The same observations about the convergence for the different configurations can be made with the hypervolume metric exposed in Table 2. Finally, we analyze the diversity of the solutions through the values of the minimal spacing metric exposed in Table 3. Once again, `APM` obtains the best value of MS with 4 islands whereas APM-C obtains the best values with 8 and 16 islands. These results show that the control islands improve the diversification capacity of APM-MOEA, especially with a large number of islands.

	4 islands		8 islands		16 islands	
	APM	APM-C	APM	APM-C	APM	APM-C
kroAB200	**0.65**	0.97	0.52	**0.48**	0.04	**0.00**
kroAB300	**0.68**	1.47	0.46	**0.37**	**0.01**	0.02
kroAB400	**0.90**	1.87	**0.38**	0.46	0.14	**0.01**
kroAB500	**1.38**	2.76	**0.42**	0.44	0.20	**0.00**
kroAB750	**1.30**	1.48	0.39	**0.34**	0.15	**0.01**
kroAB1000	**1.12**	1.15	0.42	**0.29**	0.23	**0.00**
Average	**1.00**	1.62	0.43	**0.40**	0.13	**0.01**

Table 4: Means deviations to GD for APM and APM-C

5 Conclusion and Future Work

This paper proposed a new asynchronous parallel model based on the divide-and-conquer approach to tackle multi-objective combinatorial problems. On several MOTSP instances, APM-MOEA managed to find solutions of better quality than three state-of-the-art models with 4, 8 and 16 processor cores. Moreover, APM-MOEA has proven to be scalable using from 4 to 16 islands since the computed metrics indicated good convergence and coverage of the objective space with the largest number of processors. The additional islands also allow the model to obtain a better distribution of the solutions in the search space.

Furthermore, APM-MOEA provided very promising results on one specific multi-objective problem, but future works should be dedicated to studying its behavior in other contexts. For example, it would be interesting to experiment the APM-MOEA model with continuous multi-objective or many-objective problems. Another promising avenue would be to adapt the model to other multi-objective evolutionary algorithms such as NSGA-II to further validate its efficiency.

In this study, the main focus was on studying solution quality, but master-slave models have been proposed to speed-up MOEAs. They often parallelize the evaluations of the objective functions, which usually require long computing times. Thus, it could be interesting to hybridize APM-MOEA with a master-slave model to both improve solution quality and reduce execution time.

References

1. A. Zhou, B.-Y. Qu, H. Li, S.-Z. Zhao, P. N. Suganthan, and Q. Zhang, “Multiobjective evolutionary algorithms: A survey of the state of the art,” *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 32 – 49, 2011.
2. B. S. P. Mishra, S. Dehuri, R. Mall, and A. Ghosh, “Parallel single and multiple objectives genetic algorithms: A survey,” *Int. J. Appl. Evol. Comput.*, vol. 2, no. 2, pp. 21–57, Apr. 2011.
3. F. Luna and E. Alba, “Parallel multiobjective evolutionary algorithms,” in *Springer Handbook of Computational Intelligence*, J. Kacprzyk and W. Pedrycz, Eds. Springer Berlin Heidelberg, 2015, pp. 1017–1031.
4. F. Streichert, H. Ulmer, and A. Zell, *Parallelization of Multi-objective Evolutionary Algorithms Using Clustering Algorithms*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2005, vol. 3410, book section 7, pp. 92–107.
5. A. Zinflou, C. Gagné, and M. Gravel, “GISMOO: A new hybrid genetic/immune strategy for multiple-objective optimization,” *”Computers & Operations Research ”*, vol. 39, no. 9, pp. 1951 – 1968, 2012.
6. P. C. Borges and M. P. Hansen, *A Study of Global Convexity for a Multiple Objective Travelling Salesman Problem.* Boston, MA: Springer US, 2002, pp. 129–150.

7. A. Fernandez, C. Gil, R. Baos, and M. G. Montoya, “A parallel multi-objective algorithm for two-dimensional bin packing with rotations and load balancing,” *Expert Systems with Applications*, vol. 40, no. 13, pp. 5169 – 5180, 2013.
8. A. Essabri, M. Gzara, and T. Loukil, “Parallel multi-objective evolutionary algorithm with multi-front equitable distribution,” in *Grid and Cooperative Computing. GCC 2006. Fifth International Conference*, 2006, Conference Proceedings, pp. 241–244.
9. C. Sanhueza, F. Jimenez, R. Berretta, and P. Moscato, “Pasmoqap: A parallel asynchronous memetic algorithm for solving the multi-objective quadratic assignment problem,” in *2017 IEEE Congress on Evolutionary Computation, CEC 2017, Donostia, San Sebastián, Spain, June 5-8, 2017*, 2017, pp. 1103–1110. [Online]. Available: https://doi.org/10.1109/CEC.2017.7969430
10. K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: NSGA-II,” *Evolutionary Computation, IEEE Transactions on*, vol. 6, no. 2, pp. 182–197, 2002.
11. B. Dorronsoro, G. Danoy, A. J. Nebro, and P. Bouvry, “Achieving super-linear performance in parallel multi-objective evolutionary algorithms by means of cooperative coevolution,” *Computers & OR*, pp. 1552–1563, 2013.
12. A. Atashpendar, B. Dorronsoro, G. Danoy, and P. Bouvry, “A parallel cooperative coevolutionary smpso algorithm for multi-objective optimization,” in *2016 International Conference on High Performance Computing Simulation (HPCS)*, July 2016, pp. 713–720.
13. T. Hiroyasu, M. Miki, and S. Watanabe, “The new model of parallel genetic algorithm in multi-objective optimization problems - divided range multi-objective genetic algorithm,” in *Evolutionary Computation, 2000. Proceedings of the 2000 Congress on*, vol. 1, 2000, pp. 333–340 vol.1.
14. F. de Toro Negro, J. Ortega, E. Ros, S. Mota, B. Paechter, and J. M. Martin, “Psfga: Parallel processing and evolutionary computation for multiobjective optimisation,” *Parallel Computing*, vol. 30, no. 56, pp. 721–739, 2004. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167819104000390
15. K. Deb, P. Zope, and A. Jain, *Distributed Computing of Pareto-Optimal Solutions with Evolutionary Algorithms*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2003, vol. 2632, book section 38, pp. 534–549.
16. J. Branke, H. Schmeck, K. Deb, and M. Reddy S, “Parallelizing multi-objective evolutionary algorithms: cone separation,” in *Evolutionary Computation. CEC2004. Congress on*, vol. 2, 2004, Conference Proceedings, pp. 1952–1957 Vol.2.
17. A. L. Jaimes and C. A. C. Coello, “MRMOGA: parallel evolutionary multiobjective optimization using multiple resolutions,” in *2005 IEEE Congress on Evolutionary Computation*, vol. 3, Sept 2005, pp. 2294–2301 Vol. 3.
18. A. Atashpendar, B. Dorronsoro, G. Danoy, and P. Bouvry, “A scalable parallel cooperative coevolutionary pso algorithm for multi-objective optimization,” *Journal of Parallel and Distributed Computing*, vol. 112, pp. 111 – 125, 2018, parallel Optimization using/for Multi and Many-core High Performance Computing.
19. M. Yedla, S. R. Pathakota, and T. M. Srinivasa, “Enhancing k-means clustering algorithm with improved initial centre,” *International Journal of Computer Science and Information Technologies*, pp. 121–125, 2010.
20. D. Garrett and D. Dasgupta, “Analyzing the performance of hybrid evolutionary algorithms for the multiobjective quadratic assignment problem,” in *2006 IEEE International Conference on Evolutionary Computation*, July 2006, pp. 1710–1717.
21. T. Lust and J. Teghem, *The Multiobjective Traveling Salesman Problem: A Survey and a New Approach.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 119–141.
22. A. Liefooghe, J. Humeau, S. Mesmoudi, L. Jourdan, and E.-G. Talbi, “On dominance-based multiobjective local search: design, implementation and experimental analysis on scheduling and traveling salesman problems,” *Journal of Heuristics*, vol. 18, no. 2, pp. 317–352, Apr 2012. [Online]. Available: https://doi.org/10.1007/s10732-011-9181-3
23. T. Lust and A. Jaszkiewicz, “Speed-up techniques for solving large-scale biobjective tsp,” *Comput. Oper. Res.*, vol. 37, no. 3, pp. 521–533, Mar. 2010. [Online]. Available: http://dx.doi.org/10.1016/j.cor.2009.01.005
24. “TSPLIB,” 2008, http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/.
25. A. Zinflou, C. Gagné, and M. Gravel, “Gismoo vs genetic and differential evolution algorithms in multiobjective optimization,” *Proceedings of the 9th Metaheuristics International Conference (MIC 2011)*, pp. pp S1–52–10, 2011.
26. C. Gagné and A. Zinflou, “An hybrid algorithm for the industrial car sequencing problem,” *Proceedings of the IEEE World Congress on Computational Intelligence (WCCI 2012), P, Brisbane, Australia*, 2012.
27. C. Gagné and A.Zinflou, “Solving multi-objective quadratic assignment problems using hybrid genetic/immune strategy,” *Proceedings of the 10th edition of the Metaheuristics International Conference (MIC 2013), Singapore*, 2013.

28. "ROMEO HPC Center." 2016, https://romeo.univ-reims.fr.
29. C. Grosan, M. Oltean, and D. Dumitrescu, "Performance metrics in multi-objective optimization," *Proceedings of the Conference on Applied and Industrial Mathematics (CAIM '03)*, pp. 121–125, 2003.
30. D. Van Veldhuizen and G. Lamont, "Evolutionary computation and convergence to a pareto front," in *Stanford University, California.* Morgan Kaufmann, 1998, pp. 221–228.
31. E. Zitzler and L. Thiele, *Multiobjective optimization using evolutionary algorithms - A comparative case study.* Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 292–301.
32. S. Bandyopadhyay, S. Pal, and B. Aruna, "Multiobjective gas, quantitative indices, and pattern classification," in *IEEE Trans Syst Man Cybern B Cybern*, 2004.

Quaternion simulated annealing for large-scale unconstraint continuous optimization problems

A. El Afia[1] and M. Lalaoui[2]

1. National School of Computer Science and Systems Analysis, Mohammed V University Morocco
a.elafia@um5s.net.ma
2. National School of Computer Science and Systems Analysis, Mohammed V University Morocco
med.lalaoui@yahoo.com

Abstract

Simulated annealing (SA) is a well-known stochastic local search algorithm for solving unconstrained optimization problems. It mimics the annealing process used in the metallurgy to approximate the global optimum of an optimization problem and uses the temperature to control the search. Unfortunately, the effectiveness of simulated annealing drops drastically when dealing with a large-scale optimization problem. This is due in general to a premature convergence or a stagnation. The both phenomenon's can be avoided by a good balance between exploitation and exploration. This paper focuses on the same problem encountered by simulated annealing and try to heal it using the quaternion which are a number system that extends complex numbers. Quaternion's representation helps simulated annealing algorithm to smooth the fitness landscape and thus avoiding to get stuck in the local optima by expanding the original search space. Empirical analysis was conducted on many numerical benchmark functions. The experimental results show that the quaternion's representation of neighborhood improves the quality of solution compared with the classical simulated annealing. Our approach was also compared with other nature-inspired optimization algorithms. It was shown that the quaternion simulated annealing overcomes other heuristics in terms of solution quality for most of the benchmark functions.

***Keywords**: Simulated Annealing, quaternion, local search*

1. Introduction

The optimization problem in continuous variables aims at minimizing an objective function. If this objective function depends on real variables with no restrictions on its values, the problem is called the unconstrained optimization. Otherwise, it is the constrained optimization. Mathematically, let Ω the set of feasible solution, and let $f: \Omega \to \mathbb{R}$ the objective function defined over the solution space. The purpose is to solve the unconstrained optimization problem U.O.P

$$U.O.P \quad \begin{cases} Min & f(x) \\ s.t & x \in \Omega \end{cases}$$

We need to find the global minimum x^* in the solution space Ω, where for every $x \in \ \Omega,\ f(x^*) \leq f(x)$. Let define $N(x)$ the neighborhood function for every $x \in \ \Omega$ and we consider the classical simulated annealing described in Algorithm 1. The SA starts from an initial solution $x_0 \in \ \Omega$. Then, at iteration n of the outer loop, the SA generates a new solution x_n from the previous solution x_{n-1} , this based inner loop generate x_{new} as described in algorithm 2) + the Metropolis criterion
on a probability distribution below equation (eq. 1) and decides whether or not to accept it through. This behavior is modeled by a probability distribution defined as :

$$P_{T_n}(x_{new}\ |x_{current}) = \begin{cases} exp\left(-\dfrac{f(x_{new}) - f(x_{current})}{T_{n-1}}\right) & if\ \ f(x_{new}) - f(x_{current}) > 0 \\ 1 & otherwise \end{cases} \qquad (1)$$

In which T_{n-1} is the previous temperature parameter. For each temperature stage T_n, this process is repeated L_{max} times. After these L_{max} iterations, the current temperature T_n is decreased. One of most used cooling function is the geometric function: $T_n = \alpha\, T_{n-1}\ , 0\ < \alpha < 1$
The SA repeats these steps until a stopping criterion is meet which can be the final temperature T_f or an upper limit on the number of iterations. At the early stage, the temperature is high and the probability of accepting a bad solution is large. As the algorithm proceeds, both the temperature decreases and the probability of accepting bad solution become low. At the later stage the simulated annealing behaves like the gradient descent method. In general, SA can be seen as an iterative improvement process composed from three functions: generation, acceptance and cooling. These three functions determine the convergence of general SA [1], but parameters in general SA, such as the initial temperature, initial configuration, inner-loop, and outer-loop stop criterion, can have significant impact on its finite-time behavior. That is, the computation time in practice depends on the three functions as well as these parameters. Most research on SA has concentrated on the update and accept function and various algorithmic parameters, only limited attention has been paid to the generate function.

Algorithm 1: Simulated annealing algorithm

```
Input:
$f$ : cost function, $T_0$: the initial temperature, $T_f$: the final temperature, $L_{max}$: the length of temperature stage, $x_0$ : the current solution
Output:
    $x_{best}$ : The best solution for the cost function $f$
Start:
n=0
While ($T_n \leq T_f$) do
    $x_{current} = x_n$
    For $k = 1$ to $L_{max}$ do
        $x_{new}\ = Generate\_solution(x_{current}, D, k)$
        If $(x_{New})\ -f(x_{current}) \leq 0$ then $x_{current} \leftarrow x_{new}$
            else ← Genarate a pseudorandom number $\varepsilon$ from uniform distribution over [0,1]
                If $\varepsilon < \exp(-\frac{f(x_{New} - f(x_{current}))}{T_n})$ then $x_{current} \leftarrow x_{new}$ End
        End
    End
    $\mathrm{T}_{n+1} \leftarrow \alpha\ \mathrm{T}_n$
    n←n+1
$x_{n+1} \leftarrow x_{current}$
End
$x_{best} \leftarrow x_{current}$
```

To the best of our knowledge, the first work integrating a learning technique into the neighborhood function was proposed by Corana [2], who proposed an adaptive approach to adjust the neighborhood range of SA for continuous optimization problems [More Explanation]. But It has been proved by Miki et al. 2002 [3] that this method is not better than the SA with the good neighborhood range. Miki et al, 2002 [3] tried to enhance the performance of SA by appropriately adjusting the neighborhood range according to the landscape of the given problem using the opposition based learning. In fact, the opposition based learning increases only the diversity of the candidate's solution by selection not only the random guess but also its opposite. But this approach can become when get the algorithm near to the global optimum.

Unlike the previous approaches which used parameters to adjust the neighborhood range during the exploration of the search domain, this research proposes the quaternion's representation to enhance the neighborhood exploration. In this paper we propose a method that does not use parameters to adjust neighborhood range, but instead explore the quaternion space. Each 1-dimension of the initial vector is converted to a 4-dimencsion quaternion. Even if the quaternion space is bigger than the original one the search become easier.In addition real valued data is often best understood when embedded in the complex domain.
It was first reported by Fister et al. [5] that the quaternion representation can help the algorithm to efficiently balance between exploration and exploitation. This approach expands the original search space, exploring the search space of quaternion is easier because the fitness landscape based on this quaternion representation

become smoother. The quaternion was used by Fister et al. [5] to enhance the firefly algorithm and to avoid the premature convergence; it was reported by the author that the quaternion representation of individuals within firefly creates a balance between exploration and exploitation.

Joao Papa et al. [6] introduces a Harmony search algorithm based on quaternion. His aim was smoothing the fitness landscape of non-convex function in high dimension space. Where each proposed solution in n dimensional is molded as a tensor of dimensions 4×N. The same approach was applied for swarm intelligence algorithm by Iztok Fister et al. [7] to reduce the problem of stagnation if the Bat algorithm where each individual in was represented as quaternion. In addition Thanh Tung Khuat [8] represents individuals of genetic algorithm using quaternion. The main idea behind his approach is to map each 1-dimension real value to a 4-dimensiol quaternion. The method gives a better final solution and enhance the algorithm convergences. Up to now, the quaternion was only applied on population-based algorithms. This article presents the first attempt to apply quaternion on a stochastic local search like simulated annealing and to validate its effectiveness.

The rest of this paper is organized as follows. First, section 2 gives background information, including a description of the simulated annealing and the quaternion algebra, section 3 describes the concept of the quaternion simulated annealing, then section 4 presents the experimental results and discussion, and finally, we conclude the paper in Section 5.

2. Background

In this section, first we will describe how the simulated annealing generates the candidate's solutions in the case of continuous optimization. Then, we will give a brief introduction of the quaternion algebra.

2.1. Neighborhood Structure

We suppose that for each state x in S there is a set $N(x) \subset S$, which $N(x)$ is called the set of neighbors of x reachable in exactly one move. Each move is reversible ($y \in N(x) \Longrightarrow x \in N(y)$). From any state x there is the same number of moves (i.e., $\omega = |N|$). In addition, any state in the neighborhood N must be reachable is a finite number of moves. Each move has a probability of $1/\omega$ to be accepted and each random move is choosen using the function (1) The simulated annealing explores the search space using random walk methods called the hit and run generator introduced by Smith in 1984 [9] summarized in Algorithm 2. The underling concept behind this Markov chain sampling technique is to generate a sequence of point by taking steps of random length in randomly generated direction. First the hit-and-run algorithm generate a random uniformly distributed directions over a specific set of directions on the unit hypersphere $\mathbb{R}^n$. This is done by generation n independent scalar $d_i, i = 1,2,..,n$ from a normal distribution $N(0,1)$ then we scale them to calculate the unit direction vector D_k [10]

$$D_k = (d_1, d_2 .., d_n)\left(\sum_{i=1}^{n} d_i^2\right)^{-1/2} \qquad (2)$$

Then the hit and run algorithm generates a step length λ which is generated uniformly on the intersection of unit direction vector D_k with the feasible set S.

```
Algorithm 2: $Generate\_solution()$
Input:
 $x_{current}$ : the current solution, D: the problem dimension , k : the current inner iteration index
Output:
 $x_{New}$ : The new generated solution
Start:
    $x_{k+1} = x_k + \lambda D_k$   (3)
    where $D_k$ is a random direction uniformly distributed over a direction set $D \subset S$ and $x_k$ is uniformaly distributed
     over the line set : $L_k = \{x: x \in S \; and \; x = X_k + \lambda D_k, \lambda \; a \; real \; scalar\}$
End
```

2.2. Quaternions

The quaternions [11] is defined by the formula $q = a_0 + a_1 i + a_2 j + a_3 k,$ where a_0, a_1, a_2, a_3 are real numbers i, j and k represent the imaginary parts. These fundamental quaternion units satisfy the following equations:

$$\begin{cases} ij = k, \quad jk = i \quad ki = j, \\ ji = -k \quad kj = -i \quad ik = -j \\ \quad i^2 = j^2 = k^2 = 1 \end{cases} \quad (4)$$

For two quaternions q_1, q_2 in the 4-dimential space over the real numbers, .the following operations can be defined [12]:

- Addition and subtraction are determined by the formula:

$$\begin{aligned} q_1 \pm q_2 &= (a_0 + a_1 i + a_2 j + a_3 k) \pm \ (b_0 + b_1 i + b_2 j + b_3 k) \\ &= (a_0 \pm b_0) + (a_1 \pm b_1) i + (a_2 \pm b_2) j + (a_3 \pm b_3) k \end{aligned} \quad (5)$$

- Multiplication is determined by the formula:

$$\begin{aligned} q_1 q_2 &= (a_0 + a_1 i + a_2 j + a_3 k)(b_0 + b_1 i + b_2 j + b_3 k) \\ &= a_0^1 + a_1' i + a_2' j + a_3' k \end{aligned} \quad (6)$$

Where

$$\begin{aligned} a_0' &= a_0 b_0 - a_1 b_1 - a_2 b_2 - a_3 b_3 \\ a_1' &= a_0 b_1 + a_1 b_0 + a_2 b_3 - a_3 b_2 \\ a_2' &= a_0 b_2 - a_1 b_3 + a_2 b_0 + a_3 b_1 \\ a_3' &= a_0 b_3 + a_1 b_2 - a_2 b_1 + a_3 b_0 \end{aligned}$$

The product of two quaternions is not commutative i.e $q_1 q_2 \neq \ q_2 q_1$.

- Conjugate is an unary operation defined by

$$\overline{q_1} = \overline{(a_0 + a_1 \imath + a_2 \jmath + a_3 k)} = a_0 - a_1 i - a_2 j - a_3 k \quad (7)$$

- Norm of a quaternion is determined by

$$\|q_1\| = \|a_0 + a_1 i + a_2 j + a_3 k\| = \sqrt{a_0^2 + a_1^2 + a_2^2 + a_3^2} \quad (8)$$

- The norm satisfies these proprieties $\|\overline{q_1}\| = \|q_1\|$ and $\|q_0 q_1\| = \|q_0\| \|q_1\|$. This function is used for mapping a 4-dimentional quaternion to 1-dimentional real valued scalar.
- The Multiplicative inverse of quaternion q is denoted as q^{-1} which is equal to

$$q_1^{-1} = \frac{\overline{q_1}}{\|q_1\|^2} \quad (9)$$

The multiplicative inverse of quaternion satisfies the properties

$$q_1 q_1^{-1} = q_1^{-1} q_1 = 1 \text{ , } (q_1^{-1})^{-1} = q_1 \text{ and } (q_1 q_2)^{-1} = q_2^{-1} q_2^{-1}$$

- The distance between a quaternion q_1 and q_2 is defined by

$$dist(\, q_1, \, q_2) = \sqrt{(a_0 - b_0)^2 + (a_1 - b_1)^2 + (a_2 - b_2)^2 + (a_3 - b_3)^2} \quad (10)$$

The operations of the quaternion algebra are used for implementing the quaternion hit and run function of the simulated annealing. The next section of this paper presents the use of these operations within the new simulated annealing with the quaternion hit and run generator in detail.

2.3. Simulated annealing based quaternions

The main idea behind our approach is exploring the quaternions space instead of Euclidian space. Each solution $x_n \in \mathbb{R}^D$ is modeled as a set of D quaternions $q \in \mathbb{R}^4$. Therefore, the simulated annealing based quaternion will search a solution of the form $x'_i \in \mathbb{R}^{D \times 4}$. then We map each quaternion to real value by

$$x_i = Norm(x'_i) = \frac{\left[\sum_{j=1}^{3}\left|x'_{ij}\right|\right]^{1/2}}{\left|x'_{io}\right|}, i = 1, \dots D. \qquad (11)$$

In other words, the simulated annealing tries to find the quaternions that minimizes the cost function for each variables. The simulated annealing based on the quaternion representation is relied to the classical simulated annealing, and we only changed the way we generate the candidate solution from Euclidean space to a quaternion space. The quaternion representation moves the search toward the more promising area. Despite the fact that the quaternion space is larger than the original one, it can be smoother for exploration [5]. At the first step we generate a random a set of quaternion $q^i_{initial}, i \in 1..D$ using a function defined by [5] $Random\_Quaternion() = \{a_i = N(0,1) | for\ i = 1..4\}$ (12)
Where each quaternion component is initialized with a random number drawn from the Gaussian distribution $N(0,1)$ with zero mean and one as a standard deviation. Next, each quaternion of candidate solution within the neighborhood space is mapped to the corresponding real value before evaluation the objective function.

Algorithm 3: Simulated annealing algorithm based quaternions

```
Input:
$f$ : cost function, $T_0$: the initial temperature, $T_f$: the final temperature, $L_{max}$: the length of temperature stage, $D$: problem dimension.
Output:
    $q_{best}$ : The best quaternion solution for the cost function $f$
Start:
    $q_{initial} = q_{current} = Random\_Quaternion()$
 While $(T_n \le T_f)$ do
    For $L = 1$ to $L_{max}$ do

        $q_{New} = Generate\_quaternion\_solution()$
        $N_{New} = Norm(q_{New})$
            If $(N_{New}) - f(N_{current}) \le 0$ then $q_{current} \leftarrow q_{new}$
                  else ← Genarate a pseudorandom number $\varepsilon$ from uniform distribution over [0,1]
                        If $\varepsilon < \exp(-\frac{f(N_{New}) - f(N_{current})}{T_n})$ then $q_{current} \leftarrow q_{new}$ End
            End
        End
    End
  $T_{n+1} \leftarrow \alpha T_n$
  n←n+1
 $q_{best} \leftarrow q_{current}$
End
```

In the Next section, we will study the convergence of the quaternion simulated annealing and we will prove that QSA keep its convergence propriety under the quaternion space.

3. Experimental results

Experiment was performed to find how the quaternions representation of the neighborhood can enhance the quality of solution, and improve the rate of convergence in high dimensional search space of simulated aneling. Furthermore, we compared the outcomes of our approach with other optimization algorithm such as the particle swarm optimization (PSO) [13], the genetic algorithm (GA) [14]. The ant bee colony algorithm (ABC) [15], The Bat Algorithm (BA) [16] and the generalized simulated annealing (Gen-SA) [17]. These results will have analyzed using a statistical test and then discussed.

3.1. Benchmark functions

We have chosen twelve n-dimensional functions (table 1) selected from the literature [18]. To figure the performance of our approach and how it can improve the solution quality. These functions were divided into unimodal function and multimodal one, which have multiple local minima scattered throughout the search space. These functions can show the ability of the algorithm to escape from local minimum. The n-dimensional functions can also be categorized into separable and non-separable ones The last ones are the most dificult to optimize due to the interdependency between wariables.

Table 1: The benchmark functions

No	Name	Expression	Ranges	Minimum value	Characteristics
1	**Rastrigin**	$f(\vec{x}) = 10n + \sum_{i=1}^{D}\left(x_i^2 - 10cos(2\pi x_i)\right)$	$x_i \in$ [-5.12, 5.12]	0	separable, multimodal
2	**Griewank**	$f(\vec{x}) = 1 + \sum_{i=1}^{D}\frac{x_i^2}{4000} - \prod_{I=1}^{D} cos\left(\frac{x_i}{\sqrt{i}}\right)$	$x_i \in$ [-600, 600]	0	non-separable, multimodal
3	**Rosenbrock**	$f(\vec{x}) = \sum_{i=1}^{D}\left[100\left(x_{i+1} - x_i^2\right)^2 + (1 - x_i)^2\right]$	$x_i \in$ [-5, 10]	0	non-separable, unimodal
4	**Levy**	$f(\vec{x}) = sin^2(\pi w_1) + \sum_{i=1}^{D}(w_i - 1)^2[1 + 10sin^2(\pi w_i + 1)] + (w_d - 1)^2[1 + sin^2(2\pi w_d)]$ Where $w_i = 1 + \frac{x_i}{4}$, for all $i = 1, \dots, D$	$x_i \in$ [-10, 10]	0	non-separable, multimodal
5	**Xin-She Yang**	$f(\vec{x}) = \sum_{i=1}^{D}\varepsilon_i\lvert x_i\rvert^i$ where ε is a random variable uniformly distributed in [0,1]	$x_i \in$ [-5, 5]	0	separable, multimodal
6	**Salomon**	$f(\vec{x}) = 1 - cos\left(2\pi\sqrt{\sum_{i=1}^{D}x_i^2}\right) + 0.1\sqrt{\sum_{i=1}^{D}x_i^2}$	$x_i \in$ [-100, 100]	0	non-separable, multimodal

3.2. Comparison of the convergence speed

To validate the convergence speed, experiments were conducted on the benchmarks with the dimension *D* = 10, in 10 000 neighborhood generation. The cost value was reported in figure 1. For non-separable functions $((f_2, f_3, f_4, f_6)$,it was depicted from the figures that the classical SA get stuck rapidly in local minima, it also shows a slow convergence rate in the high-dimensional search space. However, The Q-SA presents a good rate of convergence toward the global minimum especially for non-separable function. For separable functions (f_1, f_5) the Q-SA reaches a better solutions in the early stages compared to the original SA. The SA using quaternion representation converges in few generations, and it has ability to tune itself in the local minimum.

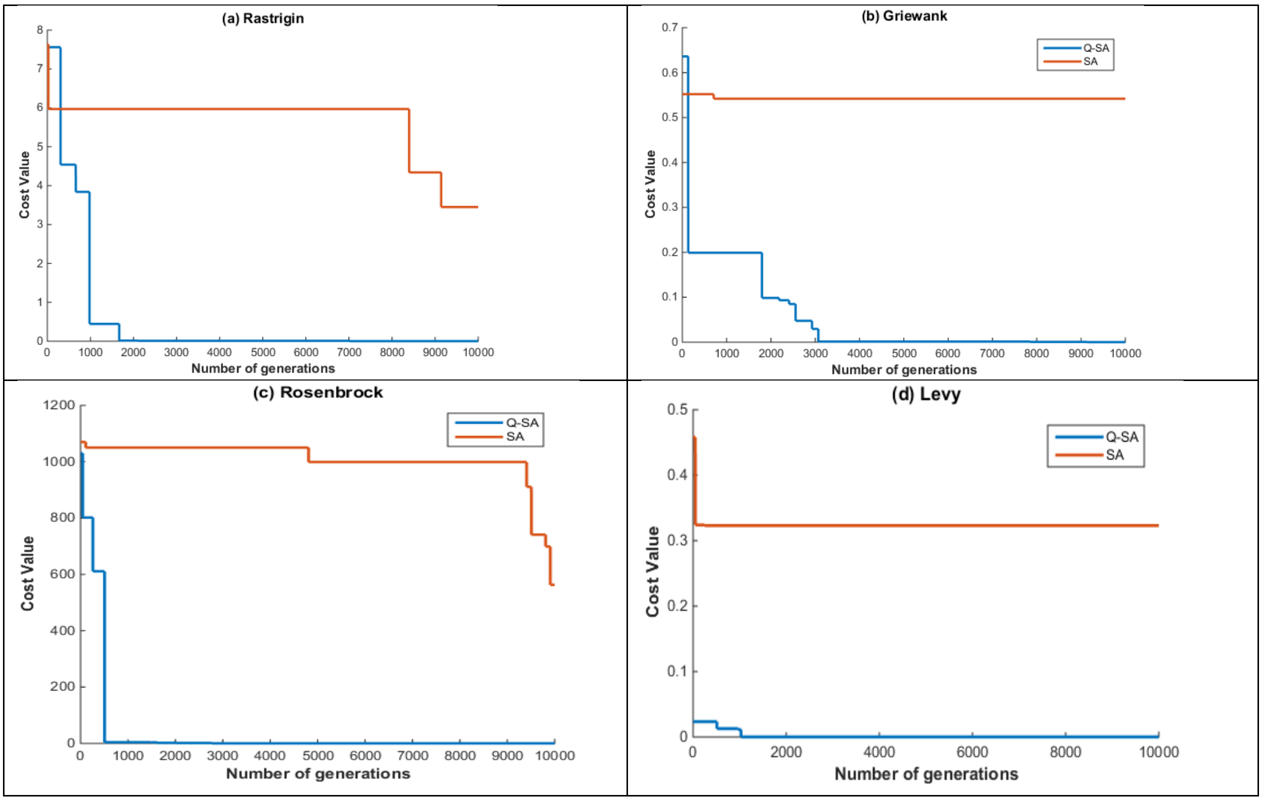


Fig 1 : Convergence curves of the Q-SA and the SA on the benchmark functions

As Observed from Fig. 1 we can find that all the cost function values of the QSA converges to much better solutions and with much faster speed at the later stage of algorithm. However, there are different convergence behaviors at the former stage for two algorithms. There are certain stages that classical SA's convergent speed outperforms QSA for some functions' plots. For example, GSA's convergent speed is faster than that of QSA at the initial stage for functions Rastrigin and Griewank. However, the convergence speed at the initial stage for functions of QSA greatly exceed CSA for All other functions. In subsequent stage, The CSA is more local than the QSA, this is why fails rapidly into a local optimum for the six functions. We can conclude that the proposed Q-SA has higher precision and tends to find the global optimum faster than the SA for all these benchmark functions.

3.3. Performance comparison with other optimization algorithms

This subsection aims to compare the result of QSA to other well-known optimization algorithms such ans the particle swarm optimization (PSO) [13], the genetic algorithm (GA) [14] ,the ant bee colony algorithm (ABC) [15],the Bat Algorithm (BA) [16] and the generalized simulated annealing (Gen-SA) [17]. This experiment part aims also to measure the effect of quaternion representation on the classical SA. The obtained results were tested using Friedman statistical tests. These experiments were conducted using an open-source library LibOPT developed in C language and implementing PSO, GA ABC and BA [19]. For the dimension D=50, we set the population size to PS=100 as reported by [5]. The maximal number of generation (MaxGen) was calculated using the following formula [5]: $MaxGen \;=\; \frac{5000\times D}{PS} \quad (12)$
Therefore, the maximal number of generation MaxGen= 2500 was used in this study for PSO, GA, ABC and BA. In addition, for each optimization algorithm a set of parameter was used.

- The Q-SA and SA parameters:
  - Initial temperature = 1000,
  - the Inner number of iteration= 100,
  - the outer number of iteration =100.
- The Gen-SA parameters was selected as [17] :
  - Initial temperature = 5230,
  - visiting.param = 2.62,
  - acceptance.param = -5.0,
  - markov.length = 2 * length(lower) lower is the vector with length of initial vector.

The parameters of GA, BA and ABC was the same used by [4]:

- The GA parameters
  - crossover rate: CR = 0.9,
  - the probability of diversity: p = 0.15,
  - the number of individuals is chosen to store in the archive pool: *m* = 40.
- The BA parameters are:
  - the loudness: *A*0 = 0.5,
  - the pulse rate: *r*0 = 0.5,
  - minimum frequency: *Qmin* = 0.0,
  - maximum frequency: *Qmax* = 0.1.
- The ABC parameters are:
  - the number of employed bees: 50,
  - the number of onlooker bees: 50,
  - the limitation of the number of cycles that a source cannot be improved: 100.
- The parameters of PSO was based on the studies of [20] :
  - the acceleration constants c1=c2= 2
  - the inertia weightw=0.7
  - the minimal inertia weightw wmin =0.4
  - the maximal inertia weightw wmax =0.9

The numerical results of our experiment for each algorithm on six benchmark functions where the best, the worst, the mean, the median values and its corresponding standard deviations for each algorithm on ten benchmark functions are presented table 3.

Table 3: Performance comparison of Q-SA with other optimization algorithms for dimension D=50.

Function	Mesures	Q-SA	SA	Gen-SA	PSO	GA	ABC	BA
Rastrigin	Best	3.06E-04	3.51E+02	3.38E+01	6.45E+01	6.88E+02	**0.00E+00**	4.08E+02
	Worst	1.29E-01	5.72E+02	8.56E+01	1.96E+02	7.93E+02	**0.00E+00**	1.07E+03
	Mean	1.41E-02	4.75E+02	5.39E+01	1.35E+02	7.40E+02	**0.00E+00**	7.81E+02
	Stdev	2.52E-02	5.65E+01	1.28E+01	2.91E+01	2.49E+01	**0.00E+00**	2.02E+02
	Median	4.82E-03	4.71E+02	5.37E+01	1.39E+02	7.41E+02	**0.00E+00**	8.19E+02
Griewank	Best	7.92E-07	2.39E+01	**3.31E-12**	8.07E-01	2.57E+01	0.00E+00	2.44E-01
	Worst	6.34E-03	3.31E+01	**4.13E-11**	1.06E+00	3.21E+01	1.34E+01	2.60E+01
	Mean	9.05E-04	2.96E+01	**1.55E-11**	9.55E-01	2.90E+01	7.02E+00	6.80E+00
	Stdev	1.46E-03	2.14E+00	**9.84E-12**	6.17E-02	1.69E+00	5.67E+00	6.62E+00
	Median	2.05E-04	2.97E+01	**1.28E-11**	9.58E-01	2.91E+01	1.02E+01	5.52E+00
Rosenbrock	Best	**1.77E-04**	3.43E+08	1.98E-13	4.00E+02	1.83E+06	4.87E+01	9.26E+01

	Worst	**4.29E+00**	4.97E+08	8.97E+01	3.84E+03	3.59E+06	1.10E+06	9.42E+05
	Mean	**2.57E-01**	4.12E+08	2.21E+01	1.87E+03	2.69E+06	3.56E+05	3.27E+04
	Stdev	**7.88E-01**	4.29E+07	2.59E+01	1.04E+03	5.30E+05	4.33E+05	1.75E+05
	Median	**6.24E-02**	4.12E+08	1.44E+01	1.63E+03	2.77E+06	4.90E+01	1.53E+02
Levy	Best	**5.95E-07**	2.19E+02	1.35E+01	5.73E+00	3.17E+02	7.93E+00	9.30E+01
	Worst	**4.55E-04**	3.36E+02	1.18E+02	4.11E+01	4.81E+02	9.56E+00	9.43E+05
	Mean	**9.03E-05**	2.79E+02	4.81E+01	1.55E+01	4.13E+02	8.83E+00	1.30E+05
	Stdev	**1.23E-04**	3.03E+01	2.02E+01	7.52E+00	4.16E+01	4.00E-01	3.31E+05
	Median	**4.13E-05**	2.79E+02	4.38E+01	1.46E+01	4.13E+02	8.95E+00	1.53E+02
Xin-She Yang	Best	3.83E-08	2.40E+11	**0.00E+00**	1.61E-10	9.86E+18	**0.00E+00**	4.79E+16
	Worst	1.41E-05	2.24E+19	**0.00E+00**	1.10E+01	1.70E+25	**0.00E+00**	9.36E+37
	Mean	2.60E-06	1.35E+18	**0.00E+00**	5.52E-01	1.16E+24	**0.00E+00**	4.26E+36
	Stdev	2.79E-06	4.20E+18	**0.00E+00**	2.07E+00	3.25E+24	**0.00E+00**	1.81E+37
	Median	1.79E-06	4.28E+16	**0.00E+00**	0.00E+00	0.00E+00	**0.00E+00**	0.00E+00
Salomon	Best	4.91E-04	2.92E+01	7.20E+00	2.00E-01	3.22E+01	**0.00E+00**	1.72E+01
	Worst	2.07E-01	3.61E+01	2.22E+01	3.50E+00	3.64E+01	**0.00E+00**	2.67E+01
	Mean	9.56E-02	3.39E+01	1.73E+01	2.02E+00	3.45E+01	**0.00E+00**	2.20E+01
	Stdev	7.30E-02	1.29E+00	2.66E+00	7.56E-01	9.67E-01	**0.00E+00**	2.20E+00
	Median	1.01E-01	3.42E+01	1.77E+01	2.20E+00	3.43E+01	**0.00E+00**	2.18E+01

As depicted in this table 3, the Q-SA algorithm solves efficiently the Rosenbrock and the Levy problems. It was also noticed that the Q-SA shows acceptable performance for Rastrigin, Griewank. Table 3 shows also that the ABC algorithm significantly outperforms the results of the other algorithms, i.e., QSA, SA, Gen-SA,PSO, GA and BA according to dimension D = 50 in Rastrigin, Salomon and Alpine functions.

3.4. Statistical test

The significance of the results was evaluated using the Friedman's test [21]. It is a non-parametric statistical test equivalent to the parametric ANOVA. The freedman's test Hypotheses are for formulated as follows:
H_0: Each ranking of the metaheuristics within each problem is equally likely, (i.e., there is no difference between them) so that for instance, the population medians are equal: $H_0 : [\mu_1 = \cdot\cdot\cdot = \mu_N]$
H_1: At least one of the metaheuristics has a different performance than at least one of the other metaheuristics; $H_0 : [\mu_1 ,\cdot\cdot\cdot, \mu_N$ $n$ot all equal]

In addition, we rank the results of the metaheuristic for each benchmark function, giving 1 to the best algorithm and 7 to the worst one. Let $\mathrm{r}(p_{ij})$ be the rank of j^{th} algorithm in k algorithm on the i^{th} function of N benchmark functions, where k is equal to 7 and N is equal to 6 in our experiment. The average ranks of algorithm were then computed, $R_j = \frac{1}{N}\sum_{i=1}^{N}\mathrm{r}(p_{ij}) \quad for\ j \in [1..7]$ as shown in table 3. The average ranks by themselves give a useful performance comparison. As depicted in table 3 the Q-SA ranks the first with the rank average of 1.83 followed by the Gen-SA with the rank average of 2.50, the ABC, PSO, BA and SA rank the third, the fourth, fifth and sixth respectively. The GA has the worst performance over all algorithm.

Table 4: The rank for all algorithms in each benchmark function and the their average rank

	Q-SA	**SA**	**Gen-SA**	**PSO**	**GA**	**ABC**	**BA**
Rastrigin	2	5	3	4	6	1	7
Griewank	2	7	1	3	6	5	4
Rosenbrock	1	7	2	3	6	5	4
Levy	1	5	4	3	6	2	7
Xin-She Yang	3	5	1	4	6	1	7
Salomon	2	6	4	3	7	1	5
Average rank (R_j)	**1.83**	**5.83**	**2.50**	**3.33**	**6.17**	**2.50**	**5.67**

The Freidman statistic is calculated by the following formula: $\chi_F^2 = \frac{12N}{k(1+k)}\left[\sum_{j=1}^{7} R_j^2 - \frac{k(k-1)^2}{4}\right] = 24.60$
Then we calculate the Iman & Danvenport [22] statistic F_F to overcome the conservative behavior of Freidman statistic $\chi_F^2 : F_F = \frac{(N-1)\chi_F^2}{N(k-1)-\chi_F^2} = 10.80$

Where the F_F statistic is distributed according to the F-distribution with $k - 1 = 6$ and $(k-1)(N-1) = 30$ degrees of freedom. $F_F = 10.80$ is greater than the critical values of $F(6,66) = 2.35$ [23]. Thus, we reject the null hypothesis at the level of significance α=0.05. Then, we conclude that the performance of all algorithms is statistically different. We can proceed with a post hoc significant test to know if algorithm i and j are different.

To do that, we used the Holm–Bonferroni [24] method. First we start by ordering the p-value $p_1, p_2, p_3.., p_{k-1}$ associated with hypotheses $H_1, H_2, H_3.., H_{k-1}$. The Holm–Bonferroni procedure reject the null hypothesis H_1 to H_{j-1} if j is the smallest integer such that $p_j > \frac{\alpha}{k-j}$, where α is the level of significance which is equal to 0.05 in this study. Within the R_j values calculated by Friedman test shown in Table xx, the Q-SA has been taken as a reference algorithm. Indicating with R_1 the rank of Q-SA, and with R_j for $j = 2, \ldots, 6$ the rank of the remaining algorithms. To calculate the p_j value for each pair of algorithm, first we compute the z_j value given by the following equation: $z_j = \frac{R_1 - R_j}{\sqrt{\frac{k(k+1)}{6N}}}$. The probability values p_j following the normal distribution $\mathrm{N}(0,1)$ have been calculated and compared to $\frac{0.05}{7-j}$. The results of the Holm–Bonferroni procedure are given in table 5. The null hypothesis is rejected when the Q-SA is compared ton GA, SA and BA. In other words, The Q-SA statistically outperforms the GA, SA and BA. However, the null hypothesis is accepted when Q-SA is compared to PSO, ABC and Gen-SA meaning that the performance of Q-SA, PSO and ABC is indistinguishable on the selected benchmark functions.

Table 5: Results of the Holm–Bonferroni procedure

$k = 7$	Q-SA vs	z_j	p_j	$\alpha/(k-j)$	Hypothesis
1	GA	-3.474	4.49E-07	8.33E-03	Rejected
2	SA	-3.207	1.14E-04	1.00E-02	Rejected
3	BA	-3.073	2.36E-04	1.25E-02	Rejected
4	PSO	-1.202	5.41E-02	1.67E-02	Accepted
5	ABC	-0.534	2.25E-01	2.50E-02	Accepted
6	Gen-SA	-0.532	3.53E-01	5.00E-02	Accepted

This results clearly indicates that not only quaternion representation enhances the classical version of simulated annealing but it can be an alternative to population based algorithm like BA, PSO and ABC for high scale problems.

4. Conclusion

This article introduces a novel approach to enhance the simulated annealing by the quaternion's representation of the neighborhood structure. Research works dealing with quaternion' for solving the optimization problem are very limited. This work proposes the quaternion's representation of neighborhood structure in the simulated annealing algorithm that associates each 1-dimensional real-valued scalar to 4-dimensional quaternion. Despite the fact that the quaternion representation enlarges the search space, exploration is more effective. This study demonstrates that problems such the premature convergence or the stagnation arisen in large scale unconstrained optimization could be reduced or even avoided using quaternion. Numerical results show that our approach enhances significantly the quality of solution in large scale problems compared to the classical simulated annealing. In addition, the QSA is competitive with other optimization algorithms. Further research should be conducted on other local search algorithms using the quaternion algebra. Furthermore, the QSA has a promising application for the real optimization problems.

References

[1] Y. Xin Yao (2007). Dynamic Neighbourhood Size in Simulated Annealing Proceedings of the IEEE Symposium on Foundations of Computational Intelligence.

[2] A.Corana, M. Marchesi, C. Martini (1987). Minimizing Multimodal Functions of Continuous Variables with the "Simulated Annealing" Algorithm, ACM Trans. on Mathematical Software, Vol. 13, No. 3, pp. 262-280.

[3] M. Miki, T. Hiroyasu, K. Ono (2002). Simulated Annealing with Advanced Adaptive Neighborhood. The Second international workshop on Intelligent systems design and application, Atlanta, GA, USA, 113–118.

[4] R. T. Haftka, Z. Gurdal (1992). Elements of Structural Optimization. Solid Mechanics and Its Applications, Vol. 11, chap. 4, p. 124.

[5] I. Fister, X.S Yang, J. Brest (2013). Modified firefly algorithm using quaternion representation, Expert Systems with Applications, 40, 7220–7230.

[6] J. Papa, D. Pereira, A. Baldassin, X.S. Yang (2016). On the Harmony Search Using Quaternions. IAPR Workshop on Artificial Neural Networks in Pattern Recognition, pp 126-137.

[7] I. Fister, J. Brest (2015). Modified bat algorithm with quaternion representation, Evolutionary Computation (CEC), IEEE Congress, Sendai, Japan.

[8] T. T. Khuat,·M. H. Le (2017). A genetic algorithm with multi-parent crossover using quaternion representation for numerical function optimization. Applied Intelligence, Vol. 46, Issue 4, pp 810–826.

[9] R. L. Smith, (1984). Efficient Monte Carlo procedures for generating points uniformly distributed over bounded regions. Operations Research, 32, 1296–1308.

[10] D.E. Knuth (1969). The Art of Computer Programming, Volume 2, Addison-Wesley.

[11] W. R. Hamilton (1853). Lectures on Quaternions. Royal Irish Academy.

[12] D. Eberly (1999). Quaternion algebra and calculus. Geometric Tools, LLC.

[13] J. Kennedy, R. C. Eberhart, Y. Shi (2001). Swarm intelligence. 1st Edition, Morgan Kaufmann.

[14] J. Koza (1992). Genetic programming: On the programming of computers by means of natural selection. MIT Press.

[15] D. Karaboga, B. Basturk (2007). A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. Journal of Global Optimization, 459-471.

[16] X.-S. Yang, A. H. Gandomi. (2012). Bat algorithm: A novel approach for global engineering optimization. Journal of Engineering Computations, Vol. 29, Issue 5, pp. 464-483.

[17] C. Tsallis and D. A. Stariolo (1996). Generalized simulated annealing. Physica A: Statistical Mechanics and its Applications, Vol. 233, pp 395–406.

[18] M. Jamil, X. S. Yang (2013). A Literature Survey of Benchmark Functions For Global Optimization Problems, Int. Journal of Mathematical Modelling and Numerical Optimization, Vol. 4, No. 2, pp. 150–194.

[19] J.P. Papa, G.H. Rosa (2017). libopt: an open-source platform for fast prototyping soft optimization techniques, Conference'17, Washington, DC, USA.

[20] F. Marini, B. Walczak (2015).Particle swarm optimization (PSO). A tutorial. Journal of Chemometrics and Intelligent Laboratory Systems. Vol. 149, Part B, pp. 153-165.

[21] M. Friedman (1937). The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance, Journal of the American Statistical Association, 32(200), pp. 675–701.

[22] R.L. Iman, J.M Davenport (1980). Approximations of the critical region of the Friedman statistic. Journal of Communications in Statistics- A Theory and Methods, Vol. 9(6), pp. 571–595.

[23] D.J. Sheskin (2004). Handbook of Parametric and Nonparametric Statistical Procedures, 3rd Edition. Chapman & Hall/CRC.

[24] S. Holm (1979). A simple sequentially rejective multiple test procedure. Scandinavian Journal of Statistics, Vol. 6, No. 2, pp. 65-70.

A new cut-based genetic algorithm for graph partitioning applied to cell formation

Menouar Boulif

Department of computer science,
University M'hamed Bougara of Boumerdès.
Avenue de l'indépendance, 35000 Boumerdès, Algeria
boumen7@gmail.com

Keywords: group technology, manufacturing cell formation, graph partitioning, graph cuts, genetic algorithms, encoding representation.

1 Introduction

Cellular Manufacturing Systems (CMS) are an industrial implementation of the Group Technology (GT) philosophy. CMS consist of dividing the manufacturing system into cells so that similar parts are processed in the same cell. Such systems are specifically designed for job shops whose production volume is average [1]. CMS have proven ability to reduce set-up times, in-process inventories, lot sizes and production equipment while improving productivity and production system mastery [16]. There are four important steps in CMS design: (1) process planning, (2) cell formation (CF), (3) machine layout and (4) cell layout. Our paper deals with CF which is a key step in CMS design.

In the last decades, the interest of researchers on CF triggered a big amount of research that can be broadly divided into the following three non-exclusive categories [1]:

1. Methods based on the part-machine incidence matrix: The part-machine incidence matrix (PMIM) is a binary matrix that indicates the set of machines used to process each part. A large number of studies concentrate on the use of this matrix, by considering that it is the most important, if not the sole, input of the problem (e.g. [13]). Such matrix-based methods generally proceed by swapping rows and/or columns of the PMIM to yield a diagonal block structure from which part families and machine cells are obtained. This category has several limitations as it takes neither the operation sequences nor the production volumes into account.
2. Methods based on similarity coefficients: McAuley [11] was the first to use the measure of similarity between machines to identify cells. He developed a mathematical coefficient that uses only PMIM information. Since his article was published, numerous papers have tried to enhance this measure by adding further inputs, including production volumes [15,10], part operational time and operation sequences [4]. The efforts in this category tend to combine data inputs from several criteria, defining the similarity coefficient as a weighted combination of the overall criteria (see [18] for a comprehensive study). However, weak justifications are given for the weighting procedure, which is an influential parameter in the derived solutions.
3. Methods based on Meta-Heuristics: CF problem's NP-completeness prompted research to focus on heuristic methods. Meta-heuristics, have attracted the most attention, leading to Tabu search approach [9], Simulated Annealing [17], Neural Network approaches [7] and Genetic algorithms (GA) [17,5,1,2]. Literature findings proved that GA based methods are very interesting research paths in comparison to other heuristics [10]. In GA based approaches, the encoding representation is the sole means that prospects the search space. We believe therefore that it must be lent more attention in research efforts. In fact, most of the published works (e.g. [17,5,2]) that use an evolutionary approach adopt the machine-to-cell integer encoding that has proven its limitation [1].

To contribute to these efforts, this paper proposes a *new* cut-based GA *encoding representation* derived from the cut-based-graph-partitioning model [12]. The proposed cut-based solving approach supposes that the number of cells is not known *a priori* and hence, it looks for the appropriate number of cells. Furthermore, this approach is more suitable to meeting the real-life production systems requirements as it uses the actual amount of product flow that is falsely estimated by binary-PMIM-based methods, and as it considers the natural constraints such as operation sequences, maximum cell size, cohabitation and non-cohabitation constraints.

The remainder of this paper is organized as follows. In section 2, a graph partitioning formulation for the MCF problem is presented. Section 3 discusses some theoretical aspects from which the cut based encoding is derived. The next section describes the proposed genetic algorithm. Section 5 presents the results obtained by applying the proposed methods on some chosen data sets, and section 5 presents our conclusions as well as our recommendations for further research.

2 Formulation

In order to be self contained, in what follows we present the formulation of the MCF problem as a cohabitation-and-non-cohabitation-constrained graph partitioning problem [12].

2.1 Input Data

(1) $M = \{M_1, M_2, ..., M_m\}$ is a set of m machines and $P = \{P_1, P_2, ..., P_p\}$ is a set of p part types.

(2) For each part type P_k (k = 1, 2, ..., p), we suppose given:

(i) A single sequence of machines to be visited by the part:
$R_k = (<M_{k,1}>, < M_{k,2}>, ..., <M_{k,sk}>)$, where: $<M_{k,t}> \in M$ $(t = 1,2, ..., s_k)$ and s_k is the number of machines in the sequence R_k.

(ii) r_k: the mean production volume of part type P_k per time unit.

(3) Constraint data :

(i) A set of machine couples SC.

(ii) Another set of machine couples SN.

(iii) An integer number N.

2.2 Flow Graph Construction

(4) For each $(P_k, M_i, M_q) \in P \times M \times M$, we denote as v_{kiq} the number of times M_i follows M_q or inversely in R_k $(i,q=1, ..., m)$.

(5) For each $(M_i, M_q) \in M \times M$, we denote the M_i, M_q inter-machine traffic $(i,q=1,..., m)$ by:

$$t_{iq} = \sum_{k=1}^{p} r_k v_{kiq}$$

In addition, we define:

(6) The non-oriented flow graph $G=(M,E)$, where the set of edges E is the set of non-ordered machine couples that are connected by a positive traffic or that are in SC or SN:

$$E = \{e_{iq} \,/\, (M_i, M_q) \in M \times M, \; i, q = 1, ..., m; \; i \neq q \text{ and } t_{iq} \neq 0\} \cup SN \cup SC$$

(7) Edge weight function W:

$$W(e_{iq}) = t_{iq} \text{ where } i,q \in \{1, ..., m\}.$$

Remark. If the flow graph is not connected, it must be connected by adding fictive edges with null weights. This procedure permits the assumption that the flow graph is connected from here on.

2.3 Decision Variables

(8) Let $CS=\{w_1,w_2,...,w_{|CS|}\}$ be a subset of cuts of G such that $\left[\frac{m}{N}\right] \leq |CS| < m$ where $[x]$ is the integer part of the number x and $|X|$ is the cardinal of the set X.

2.4 Intermediate Processing

(9) Let $C = \{C_1, C_2, ..., C_J\}$ be the set of connected components of the graph G after removing all the edges of CS cuts. That is,

$$C = \left(M, E - U_{i=1}^{|CS|} w_i\right).$$

C is a partition of M in J cells. That is,

$$C_j \neq \emptyset, \forall \; j \in \{1,2, ...,J\} \;\; ; \cup_{j=1}^{J} C_j = M \text{ and } C_j \cap Cg = \emptyset, \forall \; j,g \in \{1,2, ...,J\}, j \neq g.$$

(10) Subset of intercellular edges:

$$E(CS) = U_{i=1}^{|CS|} w_i$$

(11) Total intercellular traffic:

$$T(CS) = \sum_{e_{iq} \in E(CS)} W(e_{iq})$$

2.5 Constraints

To be feasible CS must satisfy the following constraints:

(12) Maximum number of machines allowed in a cell N: We only consider cut subsets CS whose C partition respects:

$$\forall\, C_j \in C\ (j = 1, 2, ..., J):\ |C_j| \leq N.$$

(13) Cohabitation constraint:

$$\forall\, (M_i\,,\, M_q) \in SC,\ \forall w \in CS:\ e_{iq} \notin w,\ \text{where } i,q \in \{1,..., m\}.$$

(14) Non-cohabitation constraint:

$$\forall\, (M_i,\, M_q) \in SN,\ \exists\ w \in CS:\ e_{iq} \in w,\ \text{where } i,q \in \{1,..., m\}.$$

2.6 Objective Function

(15) Let S be the set of cut subsets that respect the previous constraints. The problem is to find a solution $CS^* \in S$, such that:

$$Z\left(CS^*\right) = \underset{CS\in S}{Min}\, T\left(CS\right)$$

This means to seek a cut subset that respects all the constraints and has the minimum amount of intercellular traffic.

3 Theoretic preliminaries

Since a solution is a graph partition, it can be represented by a sum, using the boolean operator *OR* , denoted +, of cuts (A cut is a subset of edges that can be associated with a subset of vertices A for which all these edges have exclusively and exactly one endpoint in A). The solution of figure 1, for example, can be defined by the sum of two of the depicted cuts w_1,w_2,w_3.

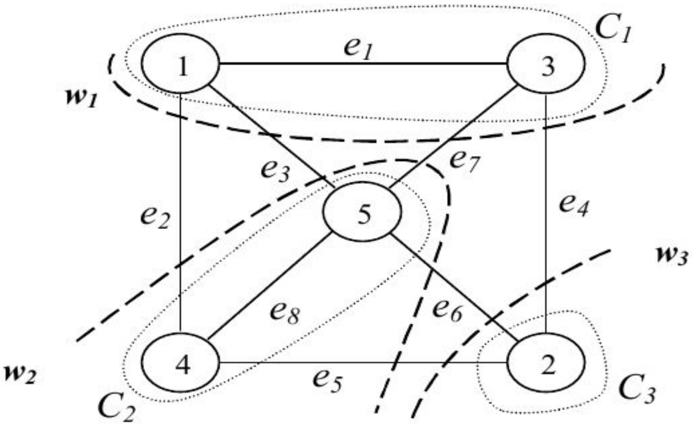


Fig. 1 A graph partition with its constructor cuts.

For instance, the sum of w_1=(0,1,1,1,0,0,1,0) and w_2=(0,1,1,0,1,1,1,0) yields w_1+w_2 =(0,1,1,1,1,1,1,0), which is sufficient to determine the associated solution. In fact, the cuts are represented by binary vectors in which the ones indicate the associated edges. For example, w_1=(0,1,1,1,0,0,1,0) is constructed by e_2,e_3,e_4,e_7 because their corresponding values equal one.

The obtained solution vector is an edge encoding representation [1]. However, the allele values used to identify *inter*cellular and *intra*cellular edges are inverted. According to this interpretation, the obtained sum (0,1,1,1,1,1,1,0) sets e_1 and e_8 intracellular and the remainder edges are intercellular, yielding the solution C={{M$_1$,M$_3$},{M$_2$},{M$_4$,M$_5$}} of figure 1.

A partition being a sum of cuts yields to the fact that the search space can be covered using these “graph creatures”. However, the edge-based cut codification is not suitable to be used as a genetic encoding representation because a random binary vector is not necessarily a cut and therefore this will require resorting to a repair function when generating the initial population or whenever a GA operator is applied. Fortunately, we can overcome this first hurdle by using cut properties in graph theory. In fact, cuts define a vector space that can be covered by the XOR operator and a subset of only m-1 special cuts (the base of the vector space). There are several manners to get a cut base. Among the simplest ways, the direct method consists of choosing any m-1 vertices and then getting the cuts associated with the singletons of the chosen vertices. For instance, for the graph of figure 1, if we choose the singletons $\{M_1\}$,$\{M_2\}$,$\{M_3\}$,$\{M_4\}$ their associated cuts, i.e. $w(\{M_1\})=(1,1,1,0,0,0,0,0)$, $w(\{M_2\})=(0,0,0,1,1,1,0,0)$, $w(\{M_3\})=(1,0,0,1,0,0,1,0)$ and $w(\{M_4\})=(0,1,0,0,1,0,0,1)$, define a cut base. Any cut has a unique representation as a XOR sum of fundamental cuts. For example, w_1= $w(\{M_1\})$ XOR $w(\{M_3\})$ and w_2= $w(\{M_1\})$ XOR $w(\{M_2\})$ XOR $w(\{M_3\})$. For further theoretic issues on cut properties see [14], [21].

4 The Cut-Based GA

Genetic algorithms are one of the famous optimization approaches that imitate natural evolutionary processes. In this section, the general principles of GA are first presented [1], followed by a description of the GA applied to the MCF problem.

4.1 Principles of the Genetic Algorithm

J.Holland [6] is considered to be the founder of the modern Genetic Algorithms. These algorithms are based on an analogy to natural selection. First, a chromosome structure is set to represent the solutions of the problem. Afterwards, an initial population of chromosomes is generated, either randomly or by using a given heuristic. Then, members of the population are selected, based on an evaluation function, called fitness. The fitness associates a value to each member according to its objective function. Genetic operators are then applied to the selected members to produce a new population generation. This process is repeated until achieving a certain stopping criterion.

Implementing genetic algorithms requires defining the following aspects:

- the structure of the genetic code used for representing solutions;
- the method for generating the initial population of solutions;
- an adaptation function to evaluate the fitness of each solution;
- the genetic operators used for producing a new generation; and
- certain control parameter values (eg. population size, number of iterations, genetic operator probabilities).

4.2 GA Implementation

4.2.1 Cut-Based Encoding The graph theory model (see section 2) allows encoding each solution by a chain of $K\times(m-1)$ binary alleles, where $K=\left[\dfrac{m}{N}\right]$ ($[x]$ denotes the integer part of x). In other words, this chain is composed of K parts of length m-1. Each part allows defining a cut by specifying the basic cuts that construct it with the XOR sum. Therefore, each one of the K parts yields a cut. Afterwards, by combining these cuts with an OR sum, we get the associated partition solution. The definition form of K is due to the fact that a typical good solution has probably a moderate number of cells which cannot be less than $\dfrac{m}{N}$ and thus, K cuts are sufficient to construct a good solution. For example, by supposing K to be equal to 3, the three-cell solution of figure 1 would be coded by the following chain:

part 1				*part 2*				*part 3*			
w({M₁})	*w({M₂})*	*w({M₃})*	*w({M₄})*	*w({M₁})*	*w({M₂})*	*w({M₃})*	*w({M₄})*	*w({M₁})*	*w({M₂})*	*w({M₃})*	*w({M₄})*
1	0	1	0	1	1	1	0	0	0	0	0

The interpretation of this chromosome structure is straightforward: part 1 uses $w(\{M_1\})$ and $w(\{M_3\})$ to define the first cut w_1. The second uses $w(\{M_1\})$, $w(\{M_2\})$ and $w(\{M_3\})$ yielding w_2. In the third part all the alleles are equal to zero, and thus, no cut is generated. The two cuts, w_1 and w_2, yield the associated partition by combining them with an OR sum.

The most important advantage of binary coding is that GAs are positively sensitive to reduced alphabets. With a binary alphabet, it becomes easier to the GA to detect the good building blocks of the individuals' codes. However, the cut-based GA suffers from a high level of redundancy. In fact, a solution is not affected by swapping its parts and, furthermore, we can have two equal parts. To overcome this second hurdle, we sort every solution without repetition. This sorting considers for each part the decimal number taken from the sub-string of the part when it is supposed binary coded (see the appendix). For example, the previous chain is coded in decimal as follows:

part 1	*part 2*	*part 3*
10	14	0

Thus, part 2 sub-string must be put first, then part1. The sorting ignores part3 because it is not associated to a valid cut. Hence, after this *sorting procedure*, there will be no equal parts except for a possible sequence-of-zeros tail.

4.2.2 Initial Population The initial population is randomly generated without repetition. To get a solution, we generate K integers from the interval [0, 2^{m-1}-1] (see the appendix). Each integer, when it is binary coded, will give a part of the solution chain. After applying the sorting procedure, if there is no equivalent member in the population, the solution is accepted.

4.2.3 Fitness and Selection To allow the GA to get advantage of the good information infeasible solutions can hide, the fitness is calculated by using a transformation function proposed by [1]. This method enables the GA to distinguish between feasible solutions and infeasible ones as well as between good and less good feasible solutions.

By using this fitness, the "Roulette wheel" random procedure [3] selects the individuals eligible to the crossover.

4.2.4 Crossover and Mutation For simplicity, we have opted for one-cutting-point crossovers. The first one is classical and allows putting the cutting point in any random point of the chain. The second crossover allows putting the cutting point only between consecutive cut parts (see figure 2). The ratio of individuals that will undergo a crossover operator is defined by the parameter *Pc*. The rest give up their places to other randomly generated members.

The mutation operator consists of randomly choosing a ratio of *Pm* members. For each one, a cut part is replaced by another one randomly generated.

For the parameter settings, empirical experimentation has been conducted to choose the parameter values that push the GA to perform at its best in a small amount of time (less than one minute).

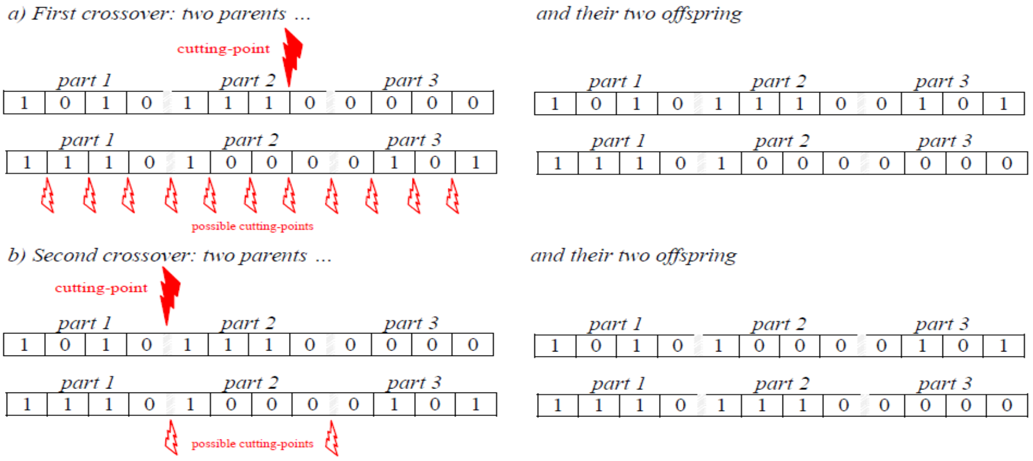


Fig. 2 Crossovers

4.2.5 Stopping Criterion After the selection-crossover-mutation process, the sorting procedure is applied on each individual of the population. The best individual that has been saved before the three step process is then reinserted in the population (elitism). This process is repeated until a certain number of iterations i_{max} is reached.

4.2.6 Cut Based GA Algorithm The pseudo code of the GA we implemented is as follows:

```
1. Get random population (with feasible or infeasible individuals without repetition); [Apply the sorting
   procedure on each individual of the population;]
2. Evaluate population using fine tuning procedure (see 3.2.3);
3. Repeat
   •   Save the best fitted individual;
```

```
    • Get mating population : Select $Pc$ proportion of individuals from population using  the Roulette Wheel
      Selection procedure;
    • Apply crossover (pick one of the two crossovers randomly) on the mating population and replace parents
      by their offspring; Generate the rest (1-$Pc$) of the population randomly;
    • Apply mutation on the resulting population with $Pm$ ratio;
    • Reinsert the best fitted individual;
    • [Apply the sorting procedure on each individual of the population;]
    • Re-evaluate population;
  Until $i_{max}$ iterations;
```

5 Computational Results

Two variants of the cut based genetic algorithm were implemented. The first one does not implement the sorting procedure whereas the second does. The two cut based GA were also compared with the edge based GA [1] and our implementation of the well-known Kmeans clustering [8]. Our implementation of Kmeans initializes the centroids by using random rows of the machine-to-machine-product-flow matrix. In addition, since Kmeans uses a known number of clusters (cells), we calls Kmeans for all the integer values in [m/N, m-1] interval; hence, we call it *multiKmeans*. The corresponding pseudo code is as follows:

```
Loop for k=[m/N] to m-1 do
    • Call kMeans(input: number of cell (clusters) k, machine-to-machine-product-flow adjacency
      matrix, output: machine partition solution);
    • update the best solution if outperformed;
Loop end;
```

The four applications were processed on a Core *i*3 microcomputer with a clock speed of 2,1 GHz and 3.8 Go of RAM managed by a 32-bit Linux operating system. We coded them by using a C++ compiler. In the following paragraphs, the four methods are referred to as CGA for the Cut based Genetic Algorithm without the sorting procedure, SCGA for the Cut based Genetic Algorithm with the Sorting procedure, EGA for the Edge based Genetic Algorithm, and multiKmeans.

Four examples are taken from the literature [1] and a fifth example has been generated randomly [20]. They are sorted according to their size, assumed to be equal to the product p×m (number of products × number of machines).

The five examples have a size of 20×8, 20×20, 40×20, 51×20 and 100×50 respectively, and the maximum number of machines per cell is set to 5 for all but the third example where it is set to 4, and the fifth where it is set to 7, then to 15.

Aiming at using moderate resources, we considered the evolutionary methods with the population sizes 100, 200, 300, 400 and 500, and the number of generations values 100, 200 and 300.

For the crossover and mutation rates, values in the interval [0.6 , 0.8] for the first, and in [0.01 , 0.05] for the second have shown comparable performances. Therefore, we have chosen to set them to 0.7 and 0.03 respectively, once and for all.

We run each one of the three methods for 20 times, and then we have reported the best average traffic and the best solution with its own computational running time. The obtained results are reported in Table 2 (the best performances for each method are boldfaced and the best scores for each example are underlined).

Table 2. Computational results.

Parameters		First example (size=20×8)									Third example (size=40×25)								
		EGA			CGA			SCGA			EGA			CGA			SCGA		
Pop. size	Gen . #	Avg. traffic	Best traffic	Cpu (s)	Avg. traffic	Best traffic	Cpu (s)	Avg. traffic	Best traffic	Cpu (s)	Avg. traffic	Best traffic	Cpu (s)	Avg. traffic	Best traffic	Cpu (s)	Avg. traffic	Best traffic	Cpu (s)
100	100	**16.0**	**13**	0.07	26.0	20	0.13	27	22	0.15	-	UF*	-	85.0	80	0.25	83.7	75	0.24
	200	19.8	17	0.13	27.6	22	0.25	24.5	21	0.29	-	UF	-	85.0	81	0.48	80.5	70	0.47
	300	25.2	19	0.19	27.4	24	0.40	25.0	20	0.44	78.1	66	0.16	84.9	81	0.73	80.1	72	0.72
300	100	18.6	15	0.22	28.6	21	0.68	**24.1**	**17**	0.77	**60.0**	51	0.19	82.7	79	0.78	74.8	68	0.82
	200	20.0	13	0.42	27.2	20	1.59	26.0	20	1.83	71.8	56	0.35	82.3	80	1.55	73.6	64	1.61
	300	19.0	13	0.65	26.0	21	2.58	27.7	22	2.56	67.2	47	0.52	82.3	80	2.36	73.4	61	2.68
500	100	17.3	13	0.40	27.4	19	1.67	25.9	22	2.57	62.1	56	0.40	80.5	**62**	1.51	75.6	68	1.60
	200	16.7	13	0.82	**25.4**	**19**	4.34	26.3	19	3.99	64.9	50	0.62	80.3	72	2.98	73.1	57	3.37
	300	18.45	13	1.21	26.0	19	6.10	26.6	19	6.56	62.0	49	0.93	**76.7**	65	4.54	**70.2**	**61**	4.98

Parameters		Second example (size=20×20)									Fourth example (size=51×20)								
		EGA			CGA			SCGA			EGA			CGA			SCGA		
Pop. size	Gen . #	Avg. traffic	Best traffic	Cpu (s)	Avg. traffic	Best traffic	Cpu (s)	Avg. traffic	Best traffic	Cpu (s)	Avg. traffic	Best traffic	Cpu (s)	Avg. traffic	Best traffic	Cpu (s)	Avg. traffic	Best traffic	Cpu (s)
100	100	30.1	27	0.06	45.9	43	0.13	38.9	38	0.11	211.2	184	0.09	192.3	166	0.44	183.7	167	0.38
	200	30.4	25	0.10	44.0	41	0.29	38.4	33	0.24	185.7	165	0.18	187.7	153	0.93	183.5	139	0.85
	300	27.5	22	0.15	41.8	38	0.49	37.0	30	0.34	199.5	168	0.26	190.6	167	1.32	180.8	145	1.25
300	100	27.7	24	0.17	40.9	38	0.53	36.2	32	0.52	202.7	162	0.29	173.0	146	2.17	169.3	139	1.81
	200	28.4	25	0.34	41.6	39	1.35	35.5	26	1.08	199.2	178	0.57	163.7	144	4.04	165.1	132	3.18
	300	26.0	22	0.54	40.8	36	1.62	34.8	31	1.55	**183.5**	**102**	0.86	164.1	145	7.3	165.3	143	5.00
500	100	26.9	22	0.30	39.5	37	1.07	34.8	31	1.04	183.8	118	0.53	170.5	152	5.36	161.7	140	3.78
	200	26.0	**20**	0.60	39.7	37	2.33	**31.6**	**25**	2.18	196.4	146	1.06	161.2	**140**	10.53	155.5	136	9.04
	300	**24.1**	21	0.90	**38.7**	**34**	3.40	33.2	25	3.19	192.7	155	1.74	**160.0**	147	16.11	**150.3**	**110**	11.52

Parameters		fifth example (size=100×35, N=7)									sixth example (size=100×50, N=15)								
		EGA			CGA			SCGA			EGA			CGA			SCGA		
Pop. size	Gen. #	Avg. traffic	Best traffic	Cpu (s)	Avg. traffic	Best traffic	Cpu (s)	Avg. traffic	Best traffic	Cpu (s)	Avg. traffic	Best traffic	Cpu (s)	Avg. traffic	Best traffic	Cpu (s)	Avg. traffic	Best traffic	Cpu (s)
100	100	-	UF	-	515.1	511	3.00	487.8	453	3.42	506.7	398	0.18	461.5	449	0.71	390.0	350	0.72
	200	-	UF	-	516.1	513	6.02	486.9	435	6.89	-	UF	-	457.3	446	1.41	378.5	342	1.48
	300	-	UF	-	513.7	503	8.84	477.7	429	9.93	-	UF	-	449.0	392	2.27	369.9	339	2.21
300	100	-	UF	-	503.0	436	9.94	477.5	447	10.34	-	UF	-	448.5	439	2.24	368.9	350	2.41
	200	529.6	465	1.49	502.2	453	18.48	463.4	435	20.37	525.8	390	1.77	434.7	363	4.51	367.9	341	5.09
	300	-	UF	-	484.7	**422**	25.19	465.1	434	31.61	522.2	415	2.40	432.0	390	7.91	363.4	**326**	7.79
500	100	530.4	482	1.36	492.2	447	15.50	466.6	432	18.12	526.1	460	1.03	435.4	**336**	4.14	364.6	338	4.40
	200	528.9	492	2.95	483.0	433	31.94	459.4	435	36.26	523.8	**349**	2.15	428.6	369	8.18	366.0	341	9.61
	300	**520.3**	**455**	3.19	**464.9**	422	45.02	**443.1**	**418**	54.56	**486.2**	394	3.78	**405.0**	345	16.16	**358.5**	331	14.83

* UF : unfeasible solution.

For the six examples, multiKmeans gave 32, 50, 194, UF, UF and 416, respectively, in less than 0.1 second.

From Table 2, in the first example, when compared to the other methods, EGA had a twofold performance: it reached a better value of the objective function in a lesser running time. In the second example, EGA was still able to reach a better value of the traffic. However, the average traffic of EGA reveals a great difficulty in reaching these best values. Indeed, CGA and SCGA were clearly more responsive to increasing the population size by reaching far better solutions in the average, as depicted in figure 3 for the fourth example.

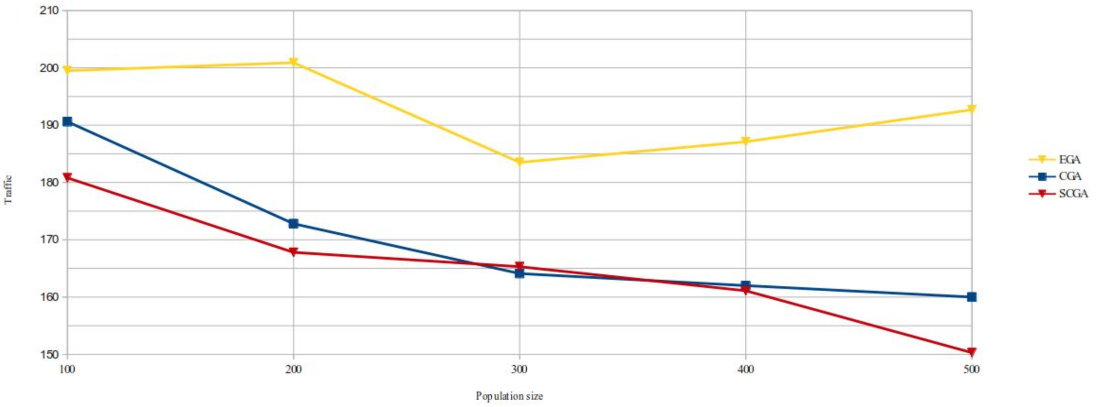
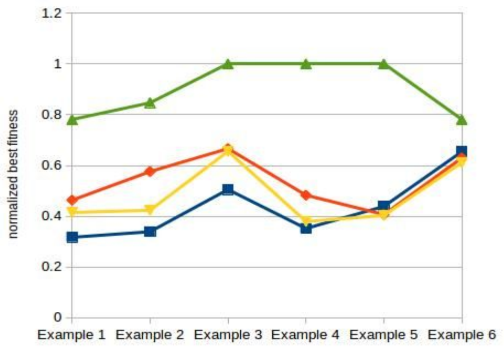


Fig. 3. Average-traffic comparison for example 4.

Figures 4 and 5 further stress the analyse by depicting the normalized values of the results, and as it can be deduced, the three evolutionary methods are far better than Kmeans based approach that does implement neither a constraint handling routine nor a mechanism to avoid local optima trapping.



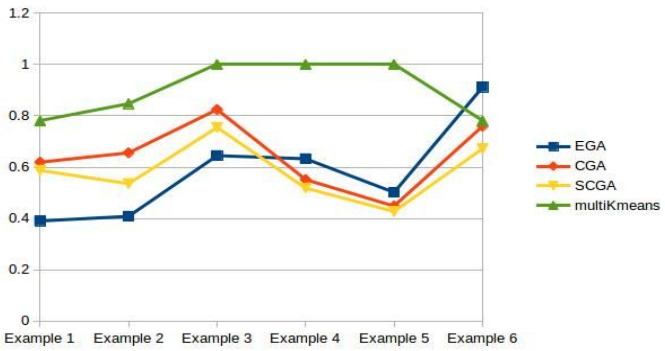


Fig. 4. Normalized-traffic comparison

Fig. 5. Normalized-average-traffic comparison

We can also realize that EGA struggles to reach feasible solutions in small population sizes and limited number of generations in comparison to its counterparts. However, SCGA that gave the best performances in average requires more time resources, especially when the expected number of cells (clusters) for good solutions grows. Indeed, this behaviour stems from the fact that when the maximum cell size decreases, cut based methods need more graph cuts to construct feasible solutions because with the overhead cost of the sorting procedure, SCGA needs more time to achieve its optimisation process.

6 Conclusions

This paper deals with cell formation witch is one of the main problems to be solved when dealing with cellular manufacturing. A genetic algorithm with a new graph-cut-based encoding representation is proposed. The performance of the new GA is tested on a set of numerical examples and compared with other methods. The cut based GA has proven to be more able to reach feasible areas with low resources, especially when the expected number of cells for good solutions is moderate.

We suggest continuing this work in the following directions. First, we are interested by adopting other ways for constructing the cut base. Inspecting then their influence on the performance of the cut based GA will be a good path of investigation. Second, the compared evolutionary methods are from the same family as they belong to the edge-based approach. This suggests a co-evolutionary solving approach is very promising. Finally, the branch and bound enhancement [1] being closer to the cut based GA than the edge based one, it seems that a hybridization of the two methods is another promising research path.

References

[1] Boulif M., Atif K. (2006). A new branch-&-bound-enhanced GA for the manufacturing cell formation problem. *Computers & Operations research*, 33:2219-2245.

[2] Darla SP., Naiju CD. (2014), Sagar PV. and Likhit BV., Optimization of Inter Cellular Movement of Parts in Cellular Manufacturing System Using Genetic Algorithm, *Research Journal of Applied Sciences, Engineering and Technology*, 7(1), 165-168.

[3] Goldberg DE. *(*1989). Genetic Algorithms in Search, Optimization and Machine Learning, *Addison Wesley, Reading, MA.*

[4] Gupta T., and Seifoddini H. (1990). Production data based similarity coefficient for machine-component grouping decision in the design of a cellular manufacturing system. 28:1247-69.

[5] Gupta YP., Gupta MC., Kumar A., Sundram C. (1995). Minimizing total intercell and intracell moves in cellular manufacturing: a genetic algorithm approach. *Int. J. Integ. Manuf.*, 8(2) 92-101.

[6] Holland JH. (1975). Adaptation in natural and artificial systems. Ann Arbor: University of Michigan Press.

[7] Kaparthi S., Suresh N. C. (1992). Machine-component cell formation in group technology: a neural network approach, *Int. J. Prod. Res.,* 30(6), 1353-1367.

[8] Lloyd SP. (1982), Least squares quantization in PCM, *IEEE Transactions on Information Theory*, 28(2): 129–137.

[9] Logendran R., Ramakrishna P. (1995), Manufacturing cell formation in the presence of Lot splitting and multiple units of same machine, *Int. J. Prod. Res.*, 33, 675-693

[10] Mahapatra SS., Pandian RS. (2008), Genetic cell formation using ratio level data in cellular manufacturing systems, *Int J Adv Manuf Technol.*, 38:630–640.

[11] McAULEY J. (1972), Machine Grouping For Efficient Production, *Production Engineer,* 52(2): 53-57

[12] Merchichi S., Boulif M. (2015), Constraint-driven exact algorithm for the manufacturing cell formation problem. *European Journal of Industrial Engineering*, 9(6):717-743.

[13] Nunkaew, W., Phruksaphanrat, B. (2013). Effective fuzzy multi-objective model based on perfect grouping for manufacturing cell formation with setup cost constrained of machine duplication. Songklanakarin Journal of Science & Technology, 35(6),715-726.

[14] Diestel R. (2010), Graph theory, Springer's Graduate texts in mathematics series, 4th edition, pp. 23-27.

[15] Seifoddini H. (1989), Single linkage vs average linkage clustering in machine cells formation applications, *Computers & Industrial Engineering,* 16, 419-426.

[16] Souilah A. (1994), Les systèmes cellulaires de production : Agencement intracellulaire. PhD dissertaion*,* University of Metz.

[17] Venugopal V., Narendran TT. (1992), Cell formation in manufacturing systems through simulated annealing: an experimental evaluation, *Eur J. Oper. Res.,* 63, 409-422.

[18] ~~Yin Y., Yasuda K. (2006), Similarity coefficient methods applied to the cell formation problem: A taxonomy~~ and review, *International Journal of Production Economics*, 101, 2, 329–352.

[19] Boulif M. (2010). Genetic algorithm encoding representations for graph partitioning problems. In Machine and Web Intelligence (ICMWI), International Conference on, 288-291. IEEE.

[20] Boulif M. (2018), A new cut-based genetic algorithm for graph partitioning applied to cell formation, supplementary material. DOI :10.6084/m9.figshare.6719906. (https://figshare.com/s/40afcf80ba563dc17214)

Appendix: Number of graph cuts and representations

1 Number of cuts

In a connected graph with m vertices there are 2^{m-1}-1 cuts. Indeed, a cut splits the graph in two complementary subsets of vertices. Since we can get 2^m subsets from a set of cardinality m, we have 2^{m-1} couples of complementary subsets. By taking away the special couple of the complete and the empty sets yields the number of cuts.

2 Representing cuts

We consider the graph of figure 1. The number of cuts is 2^{5-1}-1=15. Each cut can be derived from a combination of basic cuts that yields a unique integer number. This integer representation is very useful since it defines a one-to-one relation between the set of cuts and the set of integers in the interval [1, 2^{m-1}-1].

Table A.1 Different representations of cuts

Integer representation	Representation with basic cuts				Edge representation
	$w(\{M_1\})$	$w(\{M_2\})$	$w(\{M_3\})$	$w(\{M_4\})$	$(e_1,e_2,e_3,e_4,e_5,e_6,e_7,e_8)$
1	**1**	**0**	**0**	**0**	**(1,1,1,0,0,0,0,0)**
2	**0**	**1**	**0**	**0**	**(0,0,0,1,1,1,0,0)**
3	1	1	0	0	(1,1,1,1,1,1,0,0)
4	**0**	**0**	**1**	**0**	**(1,0,0,1,0,0,1,0)**
5	1	0	1	0	(0,1,1,1,0,0,1,0)
6	0	1	1	0	(1,0,0,0,1,1,1,0)
7	1	1	1	0	(0,1,1,0,1,1,1,0)
8	**0**	**0**	**0**	**1**	**(0,1,0,0,1,0,0,1)**
9	1	0	0	1	(1,0,1,0,1,0,0,1)
10	0	1	0	1	(0,1,0,1,0,1,0,1)
11	1	1	0	1	(1,0,1,1,0,1,0,1)
12	0	0	1	1	(1,1,0,1,1,0,1,1)
13	1	0	1	1	(0,0,1,1,1,0,1,1)
14	0	1	1	1	(1,1,0,0,0,1,1,1)
15	1	1	1	1	(0,0,1,0,0,1,1,1)

Table A.1 depicts the set of cuts with their different representations for the previous graph. Note that basic cuts (boldface typed) are powers of two.

Optimization of an Underground Water Pipeline Building using Swarm Intelligence Algorithm PSO

Dj. Bellala[1] H. Kalla[2] and L. Ourlis[3]

Laboratoire des Systmes et des Technologies de l'Information et de la Communication L@STIC
University of Batna 2, 05000 Algeria
bellala_djamel@yahoo.co.uk
hamoudi.kalla@gmail.com
ourlisl@gmail.com

1 Abstract

Particle swarm optimization is an intelligent optimization algorithm, which belongs to the famous category of optimization problems called meta-heuristics. Based on the pattern of swarm intelligence, PSO is inspired by social behavior of animals like fish and birds. This paper discusses the optimal slope of each region of the working area of the company in charge of building a water pipeline from point A to point D. Under some assumptions, the objective function, which initially depends on many variables, becomes a nonlinear function of two variables. The aim is to minimize largely the total cost of the set of entire tasks of the company project. It also describes the simulation result, which is carried out, on a two variable function with the help of a PSO.

Keywords - optimization, intelligent algorithm, Particle swarm optimization, nonlinear function.

2 Introduction

The major priority of every company, namely big business is to pursue maximum profits, secure tax breaks, and exploit cheap labor. In the case of a construction company, its major concern, inter alia, is to reduce largely the cost price of the set of the entire tasks of the project to be carried out. Thats why, special attention should be given to the project cost optimization (minimization). Depending on the complexity of the obtained mathematical model, the management committee has to find the suitable technique to solve efficiently the optimization function of the project to be performed.

3 Problem Formulation

A large company plans to build a water pipeline from point A up to point D, and must pass through three different and rugged terrain where the construction cost, differ from one region to another. In the following the description of each region according to difficulty level in terms of construction. The first region is almost a flat region, and the construction in such a region can be qualified as easy to do. The second region is rocky with varying degrees of similarity and complexity. The third region is a hill with elevation twice that of the second region. The project evaluation and studies has established; considering some assumptions; that the construction costs of both the second and the third regions are twice and three times that of the flat (first) region respectively. Since the project is a large-scale one, and by devoting the necessary resources, it is customary when it comes to fix the duration of each task to retain the one that would prove to be the least expensive. The problem is represented geometrically in figure 1 below. The horizontal distance between the extreme points A and D is approached to be 10 length units. This measurable distance is taken as a round number to simplify the calculus. Considering the point D; the highest point of the working regions; as the origin of the horizontal axis, x and y values are considered as radii of the hill region and the rocky one respectively, their values change from one point of their circumference to another. Expressing the z variable equals to $z = (10 - y)$, reduces the objective function by one variable; consequently, the building cost function is now depending only upon two variables instead of three.

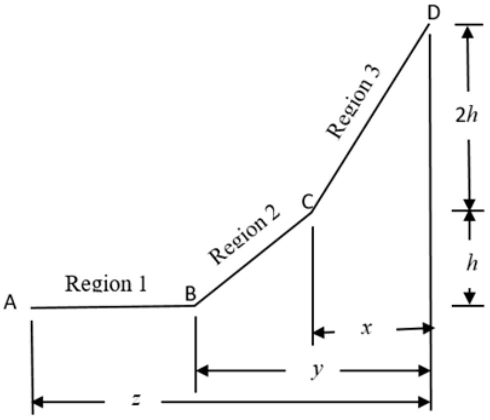


Fig. 1. Pipeline route

The total cost of the entire project is

$$Cost(x, y) = 3\sqrt{x^2 + 4} + 2\sqrt{(y - x)^2 + 1} + (10 - y) \tag{1}$$

Equation 1 represents the objective function. It is an unconstrained function, with respect to two independent variables, nonlinear and with radicals.

4 Mathematical Model of the PSO

PSO covers a population of eventual solutions called a swarm or particles. Every particle is a candidate for being a solution to the optimization problem. Within the search space, every particle takes a position. The search space is composed of all the possible solutions to the optimization problem. We would like to find the best solution among all provided ones. [1] [2] [3]
The position of every particle i is denoted by a vector $\overrightarrow{x}_i(t) \in X$ as an element of the search space X. In order to show the iteration number of the algorithm, a discrete time step t is added to this position. Thus, i is the index of the particle and $\overrightarrow{x}_i(t)$ is the position vector. The particle velocity, which is a vector of the same search space, is denoted by $\overrightarrow{v}_i(t)$. The dimensions of the position and the velocity are the same. Therefore, every particle is characterized by its time step t, its position $\overrightarrow{x}_i(t)$ and its velocity $\overrightarrow{v}_i(t)$ see figure 2. [1] [2] [3] However, this particle, which is an element of a swarm, is not acting alone; it interacts harmoniously with the rest of the other particles. They are interacting together and learning from each other according to some simple rules in order to find the best solution for the problem. The mathematical model of particles movement in the PSO is considered as a powerful tool to solve the optimization problem. $\overrightarrow{P}_i(t)$ denotes the personal best of each particle, it is the memory of its best position. $\Delta g(t)$ denotes the common best experience among the members of the swarm, it belongs to whole the swarm. All these concepts, contribute to the development of the mathematical model of the PSO, where on every iteration, position and velocity of every particle is updated according to a simple mechanism. Instantaneously the personal

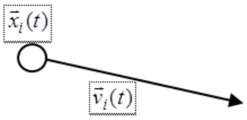


Fig. 2. Representation of a moving particle

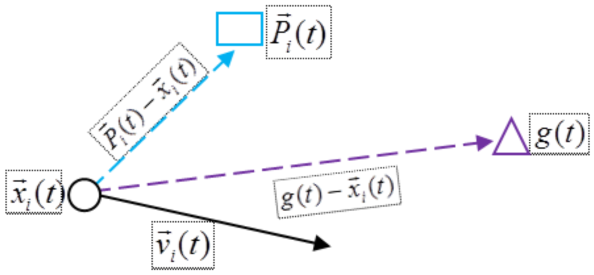


Fig. 3. Instantaneous position of a moving particle

best as well as the global best are represented with dotted vectors in figure 3.
In order to move to a new position, the particle travels parallel to the velocity vector, parallel to the vector connecting $\overrightarrow{x}_i(t)$ to $\overrightarrow{P}_i(t)$, and parallel to the vector connecting $\overrightarrow{x}_i(t)$ to $g(t)$. The result is a new position $\overrightarrow{x}_i(t+1)$ with a new velocity $\overrightarrow{v}_i(t+1)$. See figure 4. [1] [2] [3]

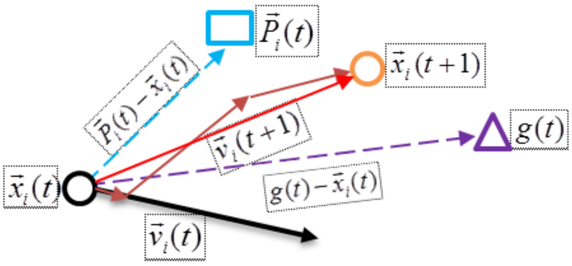


Fig. 4. New position of a moving particle

Then, the new position of each particle is created according to the previous velocity, the previous personal best and the previous global best. This new position $\overrightarrow{x}_i(t+1)$ is a candidate for being eventually a better location.

The mathematical motion model of particles in the PSO is described as follows: [1] [2] [3]

$$\begin{cases} v_i(t+1) = wv_i(t) + c_1(P_i(t) - x_i(t)) + c_2(g_i(t) - x_i(t)) \\ x_i(t+1) = x_i(t) + v_i(t+1) \end{cases} \tag{2}$$

With: w, c_1 and c_2 are real value coefficients.

The standard model of the PSO is given as follows:

$$\begin{cases} v_{ij}(t+1) = wv_{ij}(t) + r_1c_1(P_{ij}(t) - x_{ij}(t)) + r_2c_2(g(t) - x_{ij}(t)) \\ x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1) \end{cases} \tag{3}$$

with:
w: the inertia coefficient
r_1, r_2: uniformly distributed random numbers in the range $0, 1$. $r_1, r_2 \approx U(0,1)$
c_1, c_2: acceleration coefficients

The new velocity vector is composed of three components: the inertia term, the cognitive component, and the social component.

5 Structure of the basic PSO [4]

1. Initialize a population of particles with random positions and velocities;
2. Evaluate the fitness value of every particle;
3. If the fitness value is better than the previous best fitness value, then set current value as the new best;
4. Choose the particle with the best fitness value of all the particles as the global best;
5. If the stopping criterion is met, then output its objective value; otherwise, go step 2.

6 Results and Discussion

The particle swarm algorithm is implemented in Matlab system where the cost function labeled (1) has to be minimized. The cost function is with respect to two variables $(x, y) \in \Re^2$. The PSO is executed under the following parameters: [5] [6] [7]

1. The number of decision variables =2.
2. The matrix size is [1 2].
3. Range of decision variables $0 \leq x \leq y \leq 10$
4. Maximum number of iterations= 500.
5. Population size (swarm) =100.
6. Inertia coefficient =1.
7. Acceleration coefficients =2.

After initializing the position, the velocity, and the cost function of all the swarm, the algorithm converges to the following best values:

$$x = 0.707107 \qquad y = 1.284457$$

The corresponding cost is:

$$Cost(x = 0.707107, y = 1.284457) = 17.38891 \quad M.\ Units$$

7 Conclusion and Perspectives

It is verified using rules of calculus that the result is quite good. During the PSO iterations, we have seen an exponential decay of the personal best, which leads us to certify that the PSO is a global optimization algorithm, due to two factors: communication and learning.
In perspective, we intend to optimize the code of the PSO for parallel programming of a goal-programming problem and check to what extent the running time will be lowered.

References

1. Won. Y. Y. Wenwu C. Tae-Sang C. John M, Applied numerical methods using MATLAB, A John Wiley sons, Inc. 2005.
2. Federico Greco Travelling Salesman Problem, ISBN 978-953-7619-10-7 - 2008 In-teh.
3. Randy L. Haupt, Sue Ellen Haupt Practical genetic algorithms Published by John Wiley Sons, Inc., Hoboken, New Jersey. Copyright 2004.
4. Hemlata S. Urade Rahila Patel Study and Analysis of Particle Swarm Optimization: A Review 2nd National Conference on Information and Communication Technology (NCICT) 2011 Proceedings published in International Journal of Computer Applications (IJCA).
5. B. Jan, T. Vladimir, Optimization: Insights and Applications, Princeton University Press, Princeton Series in Applied Mathematics, 2005.
6. L. George, P. John, Numerical Methods using Matlab, Prentice Hall, Upper Saddle River, New Jersey 07458, 2000.
7. F. S. Hillier, G. J. Lieberman, Introduction to Operations Research, McGraw-Hill international, ninth edition 2010.

A Genetic Algorithm for selecting feature extraction strategy and data mining algorithm to optimize GPCR classification

S. Bekhouche[1] and Y. Mohamed Ben Ali[2]

1. Université Badji Mokhtar. Annaba, Algeria
Safia.bekhouche@gmail.com

2. Université Badji Mokhtar Annaba Algeria
Benaliyam2@yahoo.fr

Keywords: GA, Classification, GPCR, Machine learning algorithms, Optimization, Protein representation method.

I. Introduction

GPCRs are the largest family of membrane proteins and mediate most cellular responses to hormones and neurotransmitters, as well as being responsible for vision, olfaction and taste. At the most basic level, all GPCRs are characterized by the presence of seven membrane spanning α-helical segments separated by alternating intracellular and extracellular loop regions [1] as shown in figure 1. Therefore, they are the most intensively studied drug targets, mostly due to their substantial involvement in human pathophysiology and their pharmacological tractability [2].

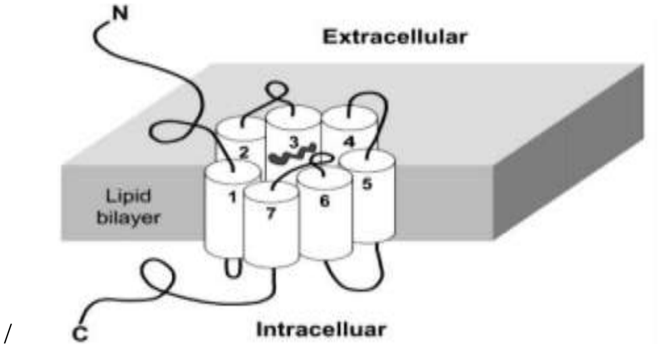


Figure1. Representation of G-protein coupled receptor structure [3]

Due to their importance and role in the human body, the identification of GPCR function is a very important task in the bioinformatics domain, that is why we must always optimize it and looking for the best possible solution.

Several works aimed at identifying the GPCRs function and improving predictive accuracy, by performing a classification step that require a numerical representation for each protein sequence made from one of the existing Feature Extraction Strategies (FES) in the literature, note that each FES produces an attribute vector of different size. For a better prediction of the function, several contributions must be taken into account among them:

- How to obtain a good accuracy and a minimal error rate of the classification?
- For each level what is the appropriate data mining algorithm to perform the classification?
- Which protein representation method will be selected for the extraction of the digital vector?
- Using a large database, does the size of the vector influences the classification results and the execution time of the selected classifiers?

In this work, we are adopted to treat the part of the classification at sub sub-family level by choosing the best FES and the convenient algorithm that result us an optimal classification with the most minimal error rate and maximum accuracy.

The bio inspired methods are still the best refuge for combinatorial problems; they can improve the results and produce the optimal solution. The FES and DMA selection is done using Genetic Algorithm (GA) that gave almost the best results for all domains [4]. Furthermore, there have been many attempts to use GAs to solve the fields of computational intelligence problems such as: Chehouri et al. [5] propose a GA to clustering applications, Djellali and Adda [6] use a new variables selection scheme based on GA meta-heuristic, also a Genetic Algorithm based approach for solving the minimum dominating set of Queens problem is described in [7],...etc.

We divide this paper into four parts: The first focuses on presentation and definition of concepts and related works according to the GPCR classification. The second is dedicated to explaining our proposed system, thus all realization steps, the third part is the experimentation for assessment of our method, and finally we terminate by some conclusions.

II. Backgrounds

Before explanation of our proposed system, we give an overview about concepts definition and works existing in this area. We divide this section into three parts, the first devoted to giving a quick view on the GPCR definition, their roles, their usefulness in the pharmaceutical domain, the second dedicated to the representation of these proteins thus a bunch of strategies used to transform an alphabetic protein sequence into a numeric vector, easy to use for any computer processing, as for the last part, We will talk about the existing classification methods to predict the GPCRs function.

2.1.GPCRs definition: Looking at the GPCRs term we can extract that it consists of two essential concepts which are: Receptors and G protein. Before defining GPCRs concept, we will give a quick view concerning these two notions.

- Receptor: It's a kind of target protein [8] (enzyme, ion channel; carrier, receptor) used to cling to ligand that can be a drug, hormone, neurotransmitter or chemical substance for give the desirable actions as a recognition site. The figure A and B show the functioning principle of the receptors thus their site in the cell.

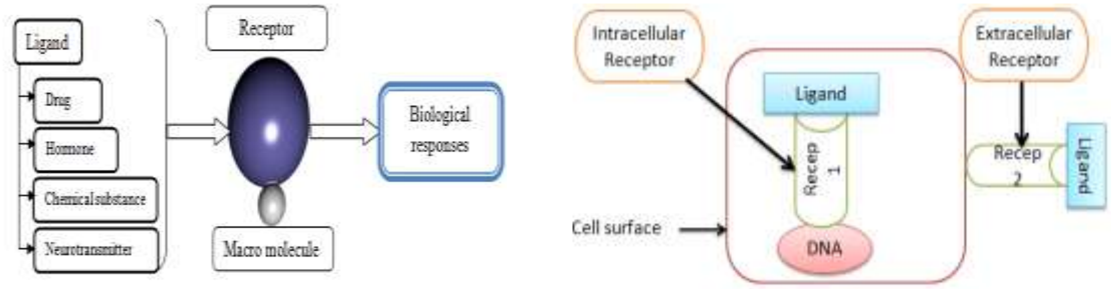


Figure1: A: Functioning principle of the receptors. (B): Site of receptors.

- **G** protein, that are so-called because they bind the **G**DP and **G**TP. They are heterotrimerics [9] owing to making of three different subunits which are: Gα, Gβ and Gγ. It allows the transfer of information inside the cell. It thus participates in a mechanism called signal transduction. To perform their functioning they must be linked to the receptors [10].

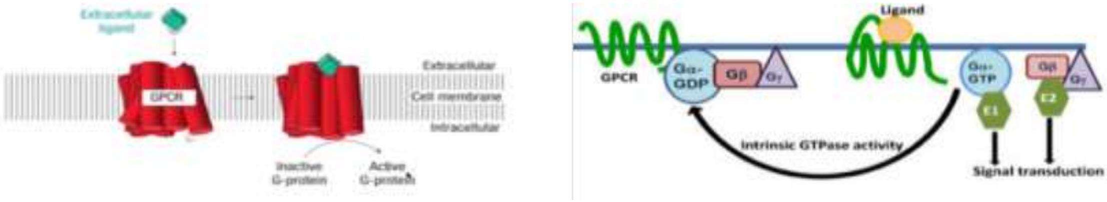


Figure 02: Signaling mechanism of heterotrimeric G-protein.

From those figures, we can remark that if the extracellular ligand bind to the GPCR in the cell membrane, this cause a confirmation change which where the inactive G-protein become the active G-protein;

Heterotrimeric G-protein will be activated if GTP instead of GDP is bound to Gα subunit. This activation can lead to multiple intracellular events trough a variety of intracellular proteins.

GPCRs represent one of the largest and most important families of multifunctional proteins known in the molecular biology of modern times, they are membrane proteins characterized by seven transmembrane (7TM) helices [9], they respond to different ligands. These proteins are very important for the understanding of human physiology and disease. The prominent role that GPCRs play in many physiological processes means that GPCR make up a large fraction of the targets of approved drugs [11] [12], and their functions are extremely various in that they regulate many physiological processes related to neurological and neurodegenerative, cardiovascular and metabolic control mechanisms. The estimated number of GPCRs in the human genome is 800, which is approximately 3% of the human genome [13]. Many efforts in the pharmacology domain are intended to understand their structures and their functions, that's why we aim to prove their applicability in this area, the following table summarize the related works.

Organ	GPCR roles	Targets for	Work
gut–brain–pancreatic	GPCR are key players in the postprandial control of metabolism and food intake	type 2 diabetes	[14]
myelinating glial cell biology	GPCRs have essential and diverse functions in the nervous system, but have only recently been implicated in this organ	remyelination in disease and after injury	[15]
epithelial prostate	GPR158 is associated with neuroendocrine transdifferentiation (NED) of epithelial prostate tumor cells, which plays a critical role in development of resistance to contemporary androgen receptor-target therapies	new prostate cancer drugs	[16]
The breast	Example protease-activated receptor1 (PAR1) is responsible for development of metastases in BCa patients. GPR116, plays an important role in cell adhesion and is found to be a novel regulator of BCa metastasis	Breast cancer (BCa)	[17] [18]
Tumor cell	PAR1 is also known to promote detachment and migration of the epithelial cancer cells, which is a key step in tumor metastases	Tumor metastases	[19], [17]
Colon	Low concentrations of recombinantly expressed KLK14 can stimulate colon cancer cell proliferation, presumably through PAR-2 receptor	Colon cancer cell proliferation	[20]

Table 1: Summary table of related work of GPCRs pharmacology.

2.2. Feature Extraction Strategies “FES”: The protein sequence can be represented by two different forms [21]: one is the sequential form where the protein is represented by a series of amino acids, this representation reflect all the information about the sequence order and length of protein but it leads to the difficulty when used in computing process, One of the most impo111rtant criteria was to formulate an effective mathematical expression that truly reflected the correlation between the intrinsic features of the sequences and the protein types to be predicted this is why the second form is created, which is the discrete form where the protein is represented by a set of discrete numbers or an attributes vector following to protein representation method.

a. Amino Acid Composition "AAC" method: It contains 20 components with each reflecting the occurrence frequency for one of the 20 native amino acids in a protein. If $P = \{A_1, A_2, A_3, ..., A_n\}$ It can be expressed by: $P' = \{ f_1, f_2, f_3, ... , f_{20}\}$ are the normalized occurrence frequencies of the 20 amino acids in S calculating using the equation 1, In spite of its simplicity, facility, the main shortcoming that all the sequence-order information is lost. Where N represents the sequence length and $N(A_i)$ is the total number of amino acid A_i represent in the sequence.

$$f(ai) = \frac{N(ai)}{N} \qquad (01)$$

b. Pseudo Amino Acid Composition "PseAAC" method : It has been rapidly and widely used in nearly all the areas of computational proteomics [22]. It uses an additional feature by varying the value of λ, which represents the rank of sequence order [23]. It preserves sequence order and sequence length information. PseAAC = *{A_1;A_2;...;A_{20};A_{20+1}...;$A_{20+\lambda}$}* The first 20 elements are the occurrence frequencies of the 20 amino acids. The remaining *A_{21},A_{22},...,$A_{20+\lambda}$* elements are first-tier to λ-tier correlation factors of amino acid sequences in the GPCR chain [24]. The components corresponding to the sequence correlation factors may be < 0,

c. Dipeptide Composition: The dipeptide components are important parameters for protein representation. $P' = \{ f_1 , f_2 , f_3 , ... , f_{400}\}$ are the absolute occurrence frequencies of the 400 dipeptides (pair of amino acid: AA, AC, .., AY, CA, CC,..,CY....YA,....YY) obtaining using equation (02).

$$f(a_i a_j) = \frac{N(a_i a_j)}{\sum_{i=1}^{20} \sum_{j=1}^{20} N(a_i a_j)} \quad (02)$$

Where $\sum_{i=1}^{20} \sum_{j=1}^{20} N(a_i a_j)$ represent the total number of all dipeptides in the sequence and $N(a_i a_j)$ is the total number of $a_i a_j$ dipeptides in the protein chain [23] [25].

2.3. Existing methods of GPCR identification: Based on pharmacological knowledge, the GPCRDB information system organizes the GPCR superfamily into a hierarchy of families, subfamilies, sub-subfamilies, and types. In all, GPCRs can be grouped into 7 classes [MUN 16] which are: Class A "Rhodopsin", Class B1 "Secretin", Class B2 "Adhesion", Class C "Glutamate", Class F "Frizzled", Class T "Taste2" and Class O "Other" that includes all 7TM receptors not belonging to any of the above classes. To arrive at such classification, several methods have been proposed in the literature, we can divide them into three categories: classification using alignment [26], flat classification [27] and hierarchical classification [24].

III. Proposed approach

In this section, we will detail our proposition and the necessary steps for its achievement. The following figure shows the general diagram of the proposed system.

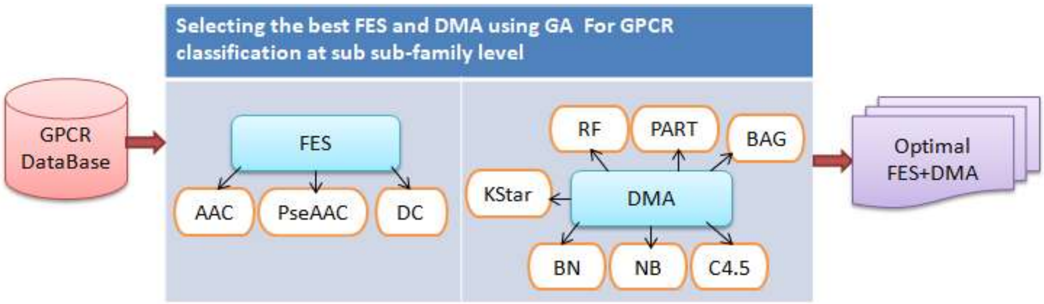

Selecting the best FES and DMA using GA For GPCR classification at sub sub-family level
GPCR DataBase
FES
AAC
PseAAC
DC
RF
PART
BAG
KStar
DMA
BN
NB
C4.5
Optimal FES+DMA

Figure 2: General architecture system.

Genetic algorithms operate on a population of individuals to produce better and better approximations. At each generation, a new population is created by the process of selecting individuals according to their level of fitness in the problem domain, and recombining them together using operators borrowed from natural genetics. The offspring might also undergo mutation. This process leads to the evolution of populations of individuals that are better suited to their environment than the individuals that they were created from, just as in natural adaptation. A state diagram for the training process with the genetic algorithm is depicted next.

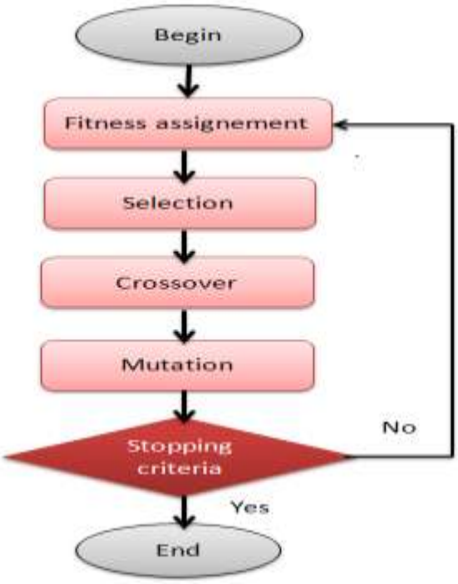


Figure3: Flow diagram of GA Search.

```
**Input** :
FES: AAC, PseAAC, DC
DMA: BN, NB, C4.5, Bag, KStar, PART, RF.
MaxIteration; ProbCross; ProbMutation;
**Output** : The best set of (FES, DMA)
Initialize Population
**While** not MaxIteration **do**
    Evaluate each individual's fitness using equation N°05.
    Selection (CrossOver & Mutation)
    Survival (Select a new Population)
**EndWhile**.
**End.**
```

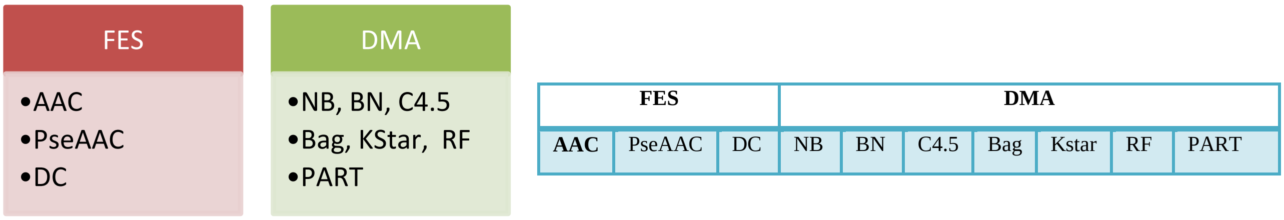
Algorithm1. GA search for selecting FES and DMA

In our case, each individual in the population represents a predictive model. The number of genes is the total number of used FES and DMA. Genes here are binary values, and represent the activation or not of particular FES, DMA in the model. The number of individuals, or population size, must be chosen for each application. Usually, this is set to be N*M, such as: N the FES number and M the DMA number.

Now we are going to describe in detail the operators and the corresponding parameters used by the genetic algorithm.

1. Initialization

The first step is to create and initialize the individuals in the population. As the genetic algorithm is a stochastic optimization method, the genes of the individuals are usually initialized at random. In order to illustrate this operator, consider a predictive model represented by figure 4. If we generate a population of 4 individuals, then we will have 4 different random FES and DMA. The next figure illustrates this population.



FES			DMA						
AAC	PseAAC	DC	NB	BN	C4.5	Bag	Kstar	RF	PART

Figure 4: Predictive model for FES and DMA selection.

1	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**

Individual 1

0	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**

Individual 2

1	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**

Individual 3

0	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**

Individual 4

As we can see, each individual is represented by 10 binary genes. The first three genes are devoted to the FES representation and the remaining are dedicated to the DMA. Each positive gene means that the corresponding Strategy/Algorithm is included in the model.

2. Fitness assignment

Once we have generated and initialized the population, we need to assign the fitness to each individual. To evaluate the fitness, we need to train the predictive model with the training data, and then evaluate its classification error rate and predictive accuracy with the selection individuals. Obviously, a high classification error rate means a low fitness. Those individuals with greater fitness will have a greater probability of being selected for recombination. The fitness value assigned to each individual will be calculated using Algorithm2:

```
Algorithm 2 : Fitness Function
Input :
FES = A chosen strategy
DMA =  A chosen classifier;
Output :
MinER = Error Rate : Calculated using equation N° 04
MaxAcc =  Accuracy : Calculated using equation N° 03
Begin
    Make a classification step according to the selected individuals.
    Compute ER, ACC using DMA;
    Compute the fitness function of each selected individual using equation N° 05.
End.
```

The following depicts the classification error rate (ER) and the accuracy (ACC). Note that the corresponding objective function of each individual is difference between the ACC measurement and the ER.

	ER	ACC	Fitness value
Individual 1	0.09	83.13%	83.04
Individual 2	0.16	90.12%	89.96
Individual 3	0.09	85.05%	84.96
Individual 4	0.04	96.34%	96.3

3. Selection

After fitness assignment has been performed, the selection operator chooses the individuals that will recombine for the next generation. The individuals most likely to survive are those more fitted to the environment. Therefore, the selection operator selects the individuals according to their fitness level. The number of selected individuals is S/2, being S the population size.

Elitism selection makes the fittest individuals to survive directly for the next generation. The elitism size controls the number of individuals which are directly selected. One of the most used selection methods is roulette wheel, that is turned and the individuals are selected at random. The corresponding individual is selected for recombination. The next figure illustrates the selection process for our example. In this case, the individual 4 has been selected by elitism, and the 3 has been selected by roulette wheel. Note that, although the individual 2 has more fitness than the 3, it has not been selected due to the stochastic nature of the genetic algorithm.

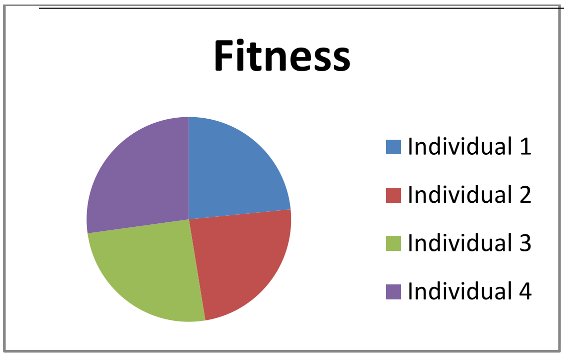


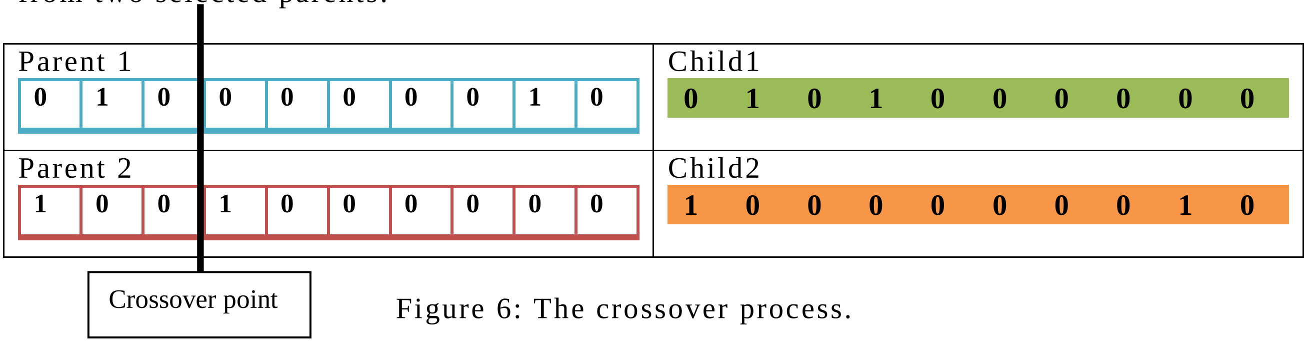
Figure 5: Selection process.

Here the number of selected individuals is half of the population size.

4. Crossover

Once the selection operator has chosen half of the population, the crossover operator recombines the selected individuals to generate a new population. This operator picks two individuals at random and combines their features to get four offspring for the new population, until the new population has the same size than the old one.

We choose the single point crossover method, fixed at point 3 of each individual. The next figure illustrates the crossover step for our example. Here we have generated two children from two selected parents.



Parent 1									
0	1	0	0	0	0	0	0	1	0
Parent 2									
1	0	0	1	0	0	0	0	0	0

Child1									
0	1	0	1	0	0	0	0	0	0
Child2									
1	0	0	0	0	0	0	0	1	0

Figure 6: The crossover process.

5. Mutation

The crossover operator can generate children that are very similar to the parents. This might cause a new generation with low diversity. The mutation operator solves this problem by changing the value of some genes in the children at random.

The mutation operator alters the characteristics of a solution in a completely random manner, which allows us to introduce and maintain diversity within our population of solutions. This operator plays the role of a "disruptive element", it introduces "noise" within the population. In order to decide if a gene will be mutated, we generate a random number between 0 and 1. If this number is lower than a value called the mutation rate, that variable is flipped. The mutation rate is a very low probability pm, generally between 0.001 and 0.01. With that value for the mutation rate, we will mutate one of each individual (statistically).

The next image shows the mutation of one of the children of the new generation. As we can see, the fourth and sixth genes of the children have been mutated.

0	1	0	0	0	1	0	0	0	0

At this point, we have the new population.

IV. Experimental part

A. Materials and methods

Tests took place on a laptop PC with an Intel(R) Core(TM) i5 processor running at 2.3 Ghz and 4 Go of RAM. Programs developed using eclipse neon environment with jdk 1.8 installed with Microsoft Windows 10 operating system.

Database: The data base that we mainly used for the training and assessment of our experimentation was downloaded from the website[1]. We did a GPCRs classification at sub sub-family level for evaluating the proposed method and looking for the best FES and DMA using GA search.

Weka: Weka (Waikato Environment for Knowledge Analysis) is a set of tools for manipulating and analyzing data files. It is written in java, available on the website[2] and implementing most of the artificial intelligence algorithms, among them, we used Naïve Bayes (NB), Bayes Net (BN), C4.5, Random Forest (RF), Meta Bagging (BAG), PART and KStar. We used weka for the assessment of the objective function for each individual using the performance measurements cited bellow.

B. Performance measurements

To measure the performance of our method, we emphasized on accuracy and error rate based classification, these measures are used to find the value of the fitness function of each solution.

The following table recapitulate all the measures used in this work thus their calculating formulas:

Abb	Description	Equation	N°
ACC	Accuracy	$Acc = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{FP} + \mathrm{FN} + \mathrm{TN}}$	03
ER	Error Rate	$ER = \frac{\text{Nombre d'exemples mal classés}}{\text{Nombre total d'exemples}}$	04
Fitness	Fitness Function	$Fitness = \mathrm{Acc} - ER$	05

Table 3. The performance measurements formulas.

Table 4 shows the progression of the fitness value variation, all results are produced with a iteration number equal to 100, the crossover probability = 0.5 and mutation probability = 0.01.

It is obvious that there is a difference between one DMA and another in each FES, and a same DMA in the used FES, for example in the AAC Strategy, the value of fitness provided by the AG is maximum and close for classifiers BN and C4.5, unlike the BAG and PART which result the poorer fitness value. As for the PseAAC method, the best fitness is carried out by the RF algorithm with 96.3% rate, the NB and KStar classifiers give close and worst results. Furthermore, the GPCR classification at sub sub-family level using DC strategy give a good and closer fitness for BN, NB, RF, BAG, PART and C4.5 DMA, so that the best fitness equal to 98.04 is achieved using PART algorithm and the RF classifier give the worst fitness which is equal to 90.56. The KStar data mining algorithm provides the null value.

FES	DMA	ACC	ER	Fitness
AAC	BN	91,35	0,11	91,24
	NB	89,15	0,13	89,02
	C4,5	90,3	0,09	90,21
	RF	88,5	0,05	88,45
	KSTAR	83,69	0,08	83,61
	BAG	82,75	0,07	82,68
	PART	82,92	0,09	82,83
PseAAC	BN	90,12	0,16	89,96
	NB	79,54	0,22	79,32
	C4,5	90,57	0,09	90,48
	RF	96,34	0,04	96,3
	KSTAR	78,65	0,09	78,56
	BAG	91,25	0,07	91,18
	PART	86,04	0,09	85,95

[1] : www.gpcrdb.org

[2] : www.cs.waikato.ac.nz/ml/weka

DC	BN	96,9	0,11	96,79
	NB	92,01	0,13	91,88
	C4,5	97	0,09	96,91
	RF	90,63	0,07	90,56
	KSTAR	0	0	0
	BAG	96,7	0,06	96,64
	PART	98,12	0,08	98,04

Table 4: Fitness value and performance measurements for DMA using different FES.

The following figure shows the progression of fitness values of each FES according to used DMA, we can extract the following points:

1. There isn't a powerful algorithm for all strategies; a DMA gives a good result with a method but not for the remaining.
2. The best algorithm used for the classification of GPCR at sub sub-family level through AAC strategy is BN that reached the 4th degree for the PseAAC method and the 3rd degree for the DC method.
3. The best algorithm used for the classification of GPCR at sub sub-family level through PseAAC strategy is RF, which reached the 4th degree for the AAC method and the 6th degree for the DC method.
4. The best algorithm used for the classification of GPCR at sub sub-family level through DC strategy is PART, which reached the 6th degree for the AAC method and the 5th degree for the PseAAC method.
5. The KStar classifier is omitted for GPCR classification using DC method because of the large number of attributes produced by this method (400 Attributes).

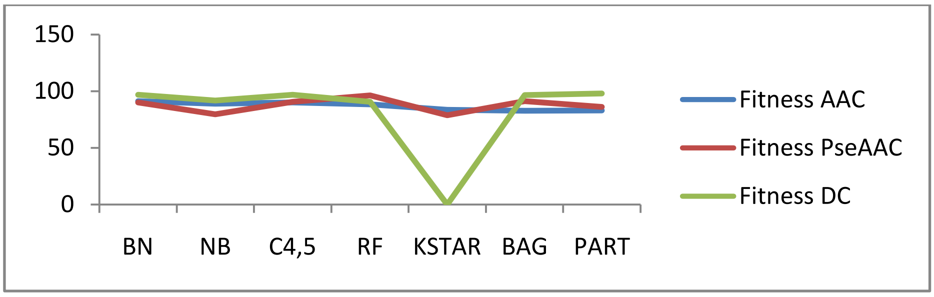


Figure 7: Progression of fitness values according to FES and DMA.

V. Conclusion and future work

As we have seen, the major problems of GPCR classification are the choice of appropriate FES for protein representation into numerical form and the convenient DMA for a good predictive accuracy also minimum error rate.

We have proposed and investigated a GA based feature extraction strategies selection and seven classifiers. A FES, DMA subsets are selected as the best for GPCR classification. The highest classification rate of 96.34% for testing set is achieved using the PseAAC method with a RF classifier. By using the statistical techniques we validated that GA for FES & DMA selection is very effective.

A further research needs to be conducted by adding more FES and DMA in the whole set for further selection on larger and several databases.

References

[1] D. M. Rosenbaum, S. G. F. Rasmussen, B. K. Kobilka, "The structure and function of G-protein-coupled receptors", Nature. pp. 356–363, May 2009.

[2] A.S. Hauser , M.M. Attwood, M. Rask-Andersen , H.B. Schiöth, D.E. Gloriam, "Trends in GPCR drug discovery: new agents, targets and indications". Nat Rev Drug Discov. pp. 829-842, Dec 2017.

[3] K.-C. Chou, D. W. Elrod," Bioinformatical analysis of G-protein-coupled receptors", Journal of proteome research, pp. 429–433, 2002.

[4] J. Huang, Y. Cai, X. Xu, "A hybrid genetic algorithm for feature selection wrapper based on mutual information", Pattern Recognition Letters, pp. 1825–1844, 2007.

[5] A. Chehouri, R. Younes, J. Khoder, J. Perron, A. Ilinca, "A Selection Process for Genetic Algorithm Using Clustering Analysis", Algorithms, 10, 123; doi:10.3390/a10040123, 2017.

[6] C. Djellali, M. Adda, "A new predictive approach to variables selection through Genetic Algorithm and Fuzzy Adaptive Resonance Theory Using medical diagnosis as a case", Procedia Computer Science, pp. 448-457, 2017.

[7] S. Alharbi, I. Venkat, "A Genetic Algorithm Based Approach for Solving the Minimum Dominating Set of Queens Problem", Journal of Optimization, vol 2017, 8 pages, Hindawi 2017.

[8] S. C. Bull, A. J. Doig, "Properties of Protein Drug Target Classes". PLoS ONE 10(3), March 2015.

[9] K.M. Kim, "Conceptual Progress for the Improvements in the Selectivity and Efficacy of G Protein-Coupled Receptor Therapeutics: An Overview". Biomol Ther . 25(1) pp. 1–3. 2017

[10] N. M. Duc, H. R. Kim and K.Y. Chung , "Recent Progress in Understanding the Conformational Mechanism of Heterotrimeric G Protein Activation". Biomol Ther 25(1), pp. 4-11. 2017

[11] J.R. Lane, A. Abdul-Ridha, M. Canals, "Regulation of G Protein-Coupled Receptors by Allosteric Ligands". Acs Chemical Neuroscience, pp. 527–534, 2013.

[12] R.T. Dorsam, J.S. Gutkind, "G-protein-coupled receptors and cancer", Nature Reviews Cancer 7, pp. 79–94. 2007.

[13] S. Siehler, G. Milligan, "G Protein-Coupled Receptors: Structure, Signaling, and Physiology", Cambridge University Press, 2010.

[14] F. Reimann, F. M. Gribble, "G protein-coupled receptors as new therapeutic targets for type 2 diabetes", Diabetologia, vol 59, pp. 229–233, 2016.

[15] A. Mogha, M. D'Rozario, K. R. Monk, "G Protein-Coupled Receptors in Myelinating Glia", Trends in Pharmacological Sciences, vol. 37, No. 11, Elsevier 2016.

[16] N. Patel, T. Itakura, S. Jeong, C-P. Liao, P. Roy-Burman, E. Zandi, S. Groshen, J. Pinski, G. A. Coetzee, M.E. Gross, M. E. Fini, "Expression and Functional Role of Orphan Receptor GPR158 in Prostate Cancer Growth and Progression", PLOS ONE | DOI:10.1371/journal.pone.0117758. February, 2015.

[17] N. A Hernández, E. Correa1, E. P Avila, T. A Vela, V. M Pérez, "PAR1 is selectively over expressed in high grade breast cancer patients: a cohort study", Journal of Translational Medicine 7:47, 2009.

[18] A. Singh, J. J. Nunes, B. Ateeq, "Role and therapeutic potential of G-protein coupled receptors in breast cancer progression and metastases", Eur J Pharmacol. 763(Pt B), pp. 178–183, Sep 2015.

[19] A. Boire , L. Covic , A. Agarwal , S. Jacques , S. Sherifi , A. Kuliopulos, "PAR1 is a matrix metalloprotease-1 receptor that promotes invasion and tumorigenesis of breast cancer cells.", Cell. 11;120(3):303-13, Feb 2005.

[20] V. Gratio, C. Loriot, G. D. Virca, K. Oikonomopoulou, F. Walker, E. P. Diamandis, M. D. Hollenberg, D. Darmoul, "Kallikrein-Related Peptidase 14 Acts on Proteinase-Activated Receptor 2 to Induce Signaling Pathway in Colon Cancer Cells", Am J Pathol. Vol. 179 N°5, pp. 2625–2636, Nov 2011.

[21] K-C. Chou, "Using amphiphilic pseudo amino acid composition to predict enzyme subfamily classes". Bioinformatics Vol. 21, pp. 10–19. 2005

[22] B. Liu, F. Yang, K-C. Chou, "2L-piRNA: A Two-Layer Ensemble Classifierfor Identifying Piwi-Interacting RNAsand Their Function". Molecular Therapy: Nucleic Acids Vol. 7. 2017.

[23] M. Naveed, A.U. Khan, "GPCR-MPredictor: multi-level prediction of G protein-coupled receptors using genetic ensemble". Amino Acids vol. 42, pp. 1809–1823. Springer, 2012.

[24] Z. ur-Rehman, A. Khan, "G-protein-coupled receptor prediction using pseudo-amino-acid composition and multiscale energy representation of different physiochemical properties". Analytical Biochemistry vol. 412, pp. 173–182. Elsevier 2011.

[25] R. Kumar, B. Kumari, M. Kumar, "Prediction of endoplasmic reticulum resident proteins using fragmented amino acid composition and support vector machine". 2017

[26] K. Harpsøe, M. W. Boesgaard, C. Munk, H. Brauner-Osborne, D. E. Gloriam, “ Structural insight to mutation effects uncover a common allosteric site in class C GPCRs. Bioinformatics”, vol 33 no 8, pp. 1116–1120. 2017

[27] A. Shkurin, A. Vellido, “Using random forests for assistance in the curation of G-protein coupled receptor databases”. BioMed Eng OnLine, vol 16 N° 75. 2017.

A new domain decomposition method for a reaction advection diffusion equation

M.R. AMATTOUCH[1] and H. Belhadj[1]

[1] University of Abdelmalek Essaadi, Faculty of sciences and techniques, department of mathematics,
BP.416, Tangier, Morocco
amattouch36@gmail.com
[2] hassan.belhadj@gmail.com

Abstract. In this paper we present an optimized domain decomposition method to resolve a reaction advection diffusion equation. We use a differential fractional derivative condition on the interface between sub-domains. We use The Fourier transform to compute and study the convergence of this method and we show that the rate of convergence of this method is zero. Then domain decomposition method operate in only two iterations. This is relevant because the method will process fast without using a preconditioner to accelerate it.
Several test-cases of analytical problems illustrate this approach and show the efficiency of the proposed new method.

Key word: equation of reaction advection diffusion, Optimized domain decomposition, Parallel computation, Caputo derivative.

1 Introduction

Many equations in the field of fluid mechanics could be modified to a reaction advection diffusion equation type. For instance, the implicit scheme of the Navier stokes equation or the linear and non linear models of turbulence could be linearized (we propose a modified fixed point method for linearization: articles [1], [2] and [3]) to these types of equations. Thus, the purpose of our work is to propose a method that is accurate with less cost for these equations. A general equation of a reaction advection diffusion equation could be:

$$\begin{cases} cu + a\frac{\partial u}{\partial x} + b\frac{\partial u}{\partial y} - \mu \Delta u = f(x,y) \;\; on\ \Omega \\ \qquad\qquad\qquad\qquad u = g \;\; on\ \partial\Omega \end{cases} \tag{1}$$

We use a domain decomposition method (DDM) to solve the equation. The principle of this method is to change a problem with large size to a decoupled consecutive sub-problems of small size that could be solved in parallel. The gain in time is significant.
We give a short story of these methods: the First mathematician to use DDM was Schwarz in order to prove the existence of harmonic functions on a complex domain [4]. After the invention of parallel softwares and computers in the seventies, a lot of theoretical works comes to use DDM for solving engineering problems, We cite S.-L. SOBOLEV [5], I. BABUSKA [6] and P.-L. LIONS [7-8], Lions was the first one to propose and prove an overlapping domain decomposition method (The Robin Robin method [9, 10]). In the nineties, we developed optimized domain decomposition methods that are faster in convergence: Especially by using Taylor Formula to approximate the steklov Poincarre operator [10,11]. In [11,12], we use The Fourier transform to approximate this operator and determine its eigenvalue: In our work we tried to perform the known optimized domain decomposition method of two order (OO2)introduced in [12]. This method is the fastest method knew in the DDM history. The OO2 method uses a specified differential interface condition between sub-domains in two order right the tangential direction and of one order in the normal direction, the method is accelerated by solving the interface condition with a preconditionneur (Krylov type, GMRES, Big Stab...).In this work we propose new differential equations in the interface somhow, our DDM converges in two iterations avoiding the use of preconditionneur that could be local for some partial differential equation. The condition of transmission in the interface between two subdomains are fractional derivative conditions.

In the first part of this work, we present the optimized decomposition domain of second order and the new proposed method. In the second part, we prove the convergence of our method and finally we give some numerical results that prove the efficiency of the proposed method.

2 Optimized domain decomposition methods

In what follows we present the principle of OO2 and our proposed method. We take for example the case of two sub-domains decomposition.
Consider the two sub-domains Ω_1 and Ω_2 with an interface Γ (see figure1)

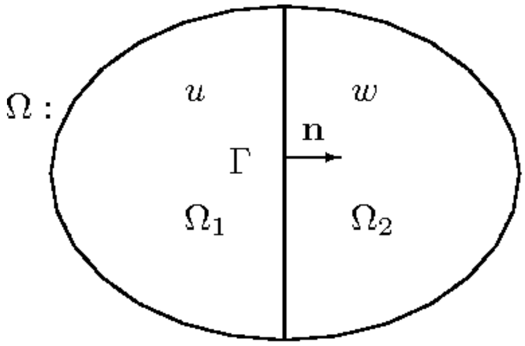


Fig. 1. splitting of the domain in two sub-domains

2.1 The OO2 method

We built two sequences u_1^p and u_2^p as follows:
Considering two initials functions u_1^0 and u_2^0 defined on Ω_1 (respectively Ω_2), we compute then u_1^p and u_2^p by solving the problems:

$$\begin{cases} L(u_1^{p+1}) = f(x,y) \ on \ \Omega_1 \\ u_1^{p+1} = \quad g \quad on \ \partial\Omega \\ B_1(u_1^{p+1}) = B_1(u_2^p) \ on \quad \Gamma \end{cases} \tag{2}$$

and

$$\begin{cases} L(u_2^{p+1}) = f(x,y) \ on \ \Omega_2 \\ u_2^{p+1} = \quad g \quad on \ \partial\Omega \\ B_2(u_2^{p+1}) = B_2(u_1^p) \ on \ \Gamma \end{cases} \tag{3}$$

where:

$$L(u) = cu + a\frac{\partial u}{\partial x} + b\frac{\partial u}{\partial y} - \mu\Delta u \tag{4}$$

and

$$\begin{array}{l} B_1(u) = \frac{\partial u}{\partial n} - C_1 u + C_2\frac{\partial u}{\partial \tau} - C_3\frac{\partial^2 u}{\partial \tau^2} \\ B_2(u) = -\frac{\partial u}{\partial n} - (C_1 - \frac{a}{\mu})u + C_2\frac{\partial u}{\partial \tau} - C_3\frac{\partial^2 u}{\partial \tau^2} \end{array}$$

n and τ are the normal and the tangent on Ω_1. B_1, B_2 are the artificial transmission condition on the interface Γ and L is the reaction advection diffusion operator of equation (1).
Because of the Fourier analysis we show that the rate of convergence of our algorithm in the fourier way is (see [12]for more explanation)

$$\rho(C_1, C_2, C_3, k) = (\frac{\lambda^-(k) - C_1 + ikC_2 + C_3k^2}{\lambda^+(k) - C_1 + ikC_2 + C_3k^2})^2 \tag{5}$$

where

$$\lambda^{\mp}(k) = \frac{a \mp \sqrt{a^2 + 4c\mu - 4i\mu bk + 4k^2\mu^2}}{2\mu}$$

are the eigenvalue of the steklov operator [12].
We don't have convergence of the OO2 method for any coefficients C_1, C_2 and C_3, but the convergence of the method is ensured if we have $\max_{|k|<\frac{\pi}{h}} \rho(C_1, C_2, C_3, k) < 1$ ([12]) and the convergence is optimized. However numerical test case show that the method is not so fast for high viscosity μ (see [1] for explanation).

2.2 A new domain decomposition method with two iteration (AlgDF)

The aim of this section is to provide a domain decomposition with fractional derivative transmission condition (AlgDF) in the interface between sub-domains. The goal of this procedure is to create a method for the reaction advection diffusion equation with a rate of convergence equal to zero, which means that our domain decomposition method is converging only by two iteration to the solution of problem (1). This method is relevant comparing to OO2 method which could make more than two iterations to converge to the solution of our problem.
In the next we also consider the splitting of our domain on two sub-domains (The general case of several domain splitting is constructed and treated by the same way). We take the notations of the section (2).

2.3 Case where a=b=0

In the case a=b=0, L is the operator that can represent the heat equation, it's eigenvalues λ^+ and λ^- are :

$$\lambda^+ = \sqrt{c + \nu k^2} \quad et \quad \lambda^- = -\sqrt{c + \nu k^2}$$

Notice that:

$$\sqrt{c + \nu k^2} = \frac{1}{\sqrt{\nu}} \sqrt{k + i\sqrt{\frac{c}{\nu}}} \times \sqrt{k - i\sqrt{\frac{c}{\nu}}}$$

This term is none other than the Fourier transform of $\frac{1}{\sqrt{\nu}} e^{-\pi \frac{c}{\nu} y} \frac{\partial^{\frac{1}{2}}}{\partial y} e^{\pi \frac{c}{\nu} y} \frac{\partial^{\frac{1}{2}}}{\partial y}$. This lead us to considerate the method with the two iterations: considering an initial function u_0 defined on Ω, we consider this next two iterations to build the solution of problem (1) on Ω_1 and Ω_2:
We solve

$$\begin{gathered} L(u_1) = f \ \ on \ \Omega_1 \\ u_1 = g \ \ on \ \partial\Omega \\ \frac{\partial u_0}{\partial n} + \frac{1}{\sqrt{\nu}} e^{-\pi \frac{c}{\nu} \tau} \frac{\partial^{\frac{1}{2}}}{\partial \tau} e^{\pi \frac{c}{\nu} y} \frac{\partial^{\frac{1}{2}} u_0}{\partial \tau} = \frac{\partial u_1}{\partial n} + \frac{1}{\sqrt{\nu}} e^{-\pi \frac{c}{\nu} \tau} \frac{\partial^{\frac{1}{2}}}{\partial \tau} e^{\pi \frac{c}{\nu} y} \frac{\partial^{\frac{1}{2}} u_1}{\partial \tau} \ \ on \ \Gamma \end{gathered}$$

Then we resolve the next iteration

$$\begin{gathered} L(u_2) = f \ \ on \ \Omega_1 \\ u_2 = g \ \ on \ \partial\Omega \\ -\frac{\partial u_0}{\partial n} + \frac{1}{\sqrt{\nu}} e^{-\pi \frac{c}{\nu}} \frac{\partial^{\frac{1}{2}}}{\partial \tau} e^{\pi \frac{c}{\nu} y} \frac{\partial^{\frac{1}{2}} u_0}{\partial \tau} = -\frac{\partial u_0}{\partial n} + \frac{1}{\sqrt{\nu}} e^{-\pi \frac{c}{\nu}} \frac{\partial^{\frac{1}{2}}}{\partial \tau} e^{\pi \frac{c}{\nu} y} \frac{\partial^{\frac{1}{2}} u_0}{\partial \tau} \ \ on \ \Gamma \end{gathered}$$

$u^{\frac{1}{2}}$ is the Caputo fractional derivative of the function u (for more detailed definitions of this operator, see for example [17]):

$$\partial^{\frac{1}{2}} u = \frac{1}{\Gamma(\frac{1}{2})} \int_0^x (x-t)^{-\frac{1}{2}} u(t) dt$$

Γ is the gamma function: $\Gamma(\frac{1}{2}) = \sqrt{\pi}$
As we construct the Fourier rate of convergence for the OO2 method, we prove by mean of the Fourier transform that the rate of convergence is null which mean that this specific domain decomposition method, operate only with two iterations to build the solution of our heat equation. To show that the rate of convergence is null we use the results bellow:

$$\begin{gathered} \mathbf{F}(\frac{\partial^{\frac{1}{2}} f}{\partial \tau}) = \sqrt{ik}\mathbf{F}(f(x)) \\ \mathbf{F}(e^{a\pi y} f) = \mathbf{F}(f)(k - ia) \end{gathered}$$

2.4 General case

In this case, a and b are arbitrary coefficients. The eigenvalues of the reaction advection diffusion operator L calculated in [12] are:

$$\lambda^{\pm} = \frac{a \pm \sqrt{4(\nu k^2 + ibk + c) - a^2)}}{2\nu} = \frac{a}{\nu} \pm \sqrt{k^2 + ibk + c + (\frac{a}{\nu})^2}$$

As we have done for the case a=b=0, we build a domain decomposition method with fractional derivative condition transmission in the interface between the two sub-domains AlgDF that provide the solution of our problem solution (1) only by two iterations: there exist two real numbers α and β such that:

$$\sqrt{k^2 + ibk + c + (\frac{a}{\nu})^2} = \sqrt{k + i\alpha} \times \sqrt{k + i\beta}$$

The two iterations of the domain decomposition method that we propose is then:

$$\begin{gathered} L(u_1) = f \;\; on \;\; \Omega_1 \\ u_1 = g \;\; on \;\; \partial\Omega \\ \tfrac{\partial u_0}{\partial n} + \tfrac{1}{\sqrt{\nu}} e^{-\pi\alpha\tau} \tfrac{\partial^{\frac{1}{2}}}{\partial\tau} e^{\pi\beta y} \tfrac{\partial^{\frac{1}{2}} e^{(\pi(\beta-\alpha))} u_0}{\partial\tau} = \tfrac{\partial u_1}{\partial n} + \tfrac{1}{\sqrt{\nu}} e^{-\pi\alpha\tau} \tfrac{\partial^{\frac{1}{2}}}{\partial\tau} e^{\pi\beta y} \tfrac{\partial^{\frac{1}{2}} e^{(\pi(\beta-\alpha))} u_1}{\partial\tau} \;\; on \;\; \Gamma \end{gathered}$$

then, we solve the next iteration problem:

$$\begin{gathered} L(u_2) = f \;\; on \;\; \Omega_1 \\ u_2 = g \;\; on \;\; \partial\Omega \\ -\tfrac{\partial u_0}{\partial n} + \tfrac{1}{\sqrt{\nu}} e^{-\pi\alpha} \tfrac{\partial^{\frac{1}{2}}}{\partial\tau} e^{\pi\beta y} \tfrac{\partial^{\frac{1}{2}} e^{(\pi(\beta-\alpha))} u_0}{\partial\tau} = -\tfrac{\partial u_2}{\partial n} + \tfrac{1}{\sqrt{\nu}} e^{-\pi\alpha} \tfrac{\partial^{\frac{1}{2}}}{\partial\tau} e^{\pi\beta y} \tfrac{\partial^{\frac{1}{2}} e^{(\pi(\beta-\alpha))} u_2}{\partial\tau} \;\; on \;\; \Gamma \end{gathered}$$

3 convergence of the AlgDF

The AlgDF consist of solving in parallel the two iterations:

$$\begin{gathered} L(u_1) = f \;\; on \;\; \Omega_1 \\ u_1 = g \;\; on \;\; \partial\Omega \\ \tfrac{\partial u_0}{\partial n} + \tfrac{1}{\sqrt{\nu}} e^{-\pi\alpha\tau} \tfrac{\partial^{\frac{1}{2}}}{\partial\tau} e^{\pi\beta y} \tfrac{\partial^{\frac{1}{2}} e^{(\pi(\beta-\alpha))} u_0}{\partial\tau} = \tfrac{\partial u_1}{\partial n} + \tfrac{1}{\sqrt{\nu}} e^{-\pi\alpha\tau} \tfrac{\partial^{\frac{1}{2}}}{\partial\tau} e^{\pi\beta y} \tfrac{\partial^{\frac{1}{2}} e^{(\pi(\beta-\alpha))} u_1}{\partial\tau} \;\; on \;\; \Gamma \end{gathered} \tag{6}$$

and

$$\begin{gathered} L(u_2) = f \;\; on \;\; \Omega_1 \\ u_2 = g \;\; on \;\; \partial\Omega \\ -\tfrac{\partial u_0}{\partial n} + \tfrac{1}{\sqrt{\nu}} e^{-\pi\alpha} \tfrac{\partial^{\frac{1}{2}}}{\partial\tau} e^{\pi\beta y} \tfrac{\partial^{\frac{1}{2}} e^{(\pi(\beta-\alpha))} u_0}{\partial\tau} = -\tfrac{\partial u_2}{\partial n} + \tfrac{1}{\sqrt{\nu}} e^{-\pi\alpha} \tfrac{\partial^{\frac{1}{2}}}{\partial\tau} e^{\pi\beta y} \tfrac{\partial^{\frac{1}{2}} e^{(\pi(\beta-\alpha))} u_2}{\partial\tau} \;\; on \;\; \Gamma \end{gathered} \tag{7}$$

where u_0 is an initial function.
Theorem:
The solutions of equations (6) and (7) are the restriction of the solution of equation (1) on Ω_1 and Ω_2 respectively.
Proof.
We explain the demonstration in the case of which the interface is the line:

$$\Gamma : x = 0 \quad (n = \overrightarrow{i} \; and \; \tau = \overrightarrow{j})$$

we get by substraction

$$L(u_1 - u) = 0 \; and \; L(u_2 - u) = 0 B_1(u_1 - u) = B_1(u_0 - u) \; and \; B_2(u_2 - u) = B_2(u_0 - u)$$

Where u is the exact solution of the problem (1) and

$$B_1(u) = \frac{\partial .}{\partial n} + \frac{1}{\sqrt{\nu}} e^{-\pi\alpha\tau} \frac{\partial^{\frac{1}{2}}}{\partial\tau} e^{\pi\beta y} \frac{\partial^{\frac{1}{2}} e^{(\pi(\beta-\alpha))} .}{\partial\tau}$$

$$B_2(.) = -\frac{\partial .}{\partial n} + \frac{1}{\sqrt{\nu}} e^{-\pi\alpha\tau} \frac{\partial^{\frac{1}{2}}}{\partial \tau} e^{\pi\beta y} \frac{\partial^{\frac{1}{2}} e^{(\pi(\beta-\alpha))}.}{\partial \tau}$$

If ω^1 and ω_1^0 is the Fourier transform of $u_1 - u$ and $u_0 - u$ respectively on Ω_1, we deduce:

$$c\omega + a\frac{\partial \omega^1}{\partial x} - ibk\omega^1 - \nu\frac{\partial^2 \omega^1}{\partial x^2} + k^2\nu\omega^1 = 0$$

k is the Fourier frequency. If we take a solution in the form of $\omega = \omega(0,k)e^{\lambda x}$ we obtain:

$$c + a\lambda - ibk - \nu\lambda^2 + \nu k^2 = 0$$

It's an equation of two order which have two solutions:

$$\lambda^{\bar{+}}(k) = \frac{a\bar{+}\sqrt{a^2 + 4c\nu - 4i\nu bk + 4k^2\nu^2}}{2\nu}$$

since $\lim\limits_{x\to-\infty} \omega^1(x,k) = 0$ we take $\omega^1(x,k) = \omega^1(0,k)e^{\lambda^+ x}$(remark that $\lambda^+ > o$ and $\lambda^- < o$)
by the same way if ω^2 and ω^0 is the Fourier transform of $u_2 - u$ and $u_0 - u$ on Ω_2, we have: $\omega^2(x,k) = \omega^2(0,k)e^{\lambda^- x}$
The two conditions:

$$B_1(\omega^1(0,k)) = B_1(\omega_1^0(0,k)) \; and \; B_2(\omega^2(0,k)) = B_2(\omega_2^0(0,k))$$

lead to:

$$0 = (\lambda^- - \lambda^-)\omega_1^0(0,k) = (\lambda^- - \lambda^+)\omega_1(0,k) \; and \; 0 = (\lambda^+ - \lambda^+)\omega_1^0(0,k) = (\lambda^+ - \lambda^+)\omega_2(0,k)$$

we get: $\omega^1 = 0$ and $\omega^2 = 0$ and then the solution $u_1 = u$ in the Fourier way on Ω_1 and $u_2 = u$ in the Fourier way on Ω_2 and this gives the result of our theorem.
Remark:
The problem (6) and (7) has one unique solution (we can prove it by Lax Milgram theorem [17, 18]) that make our method consistent and even when the coefficient c is zero or the boundary condition of problem (1)is a Neumann condition instead of the dirichlet condition.

4 Numerical results

To show Numerically the efficiency of our method compared to the OO2 method, we conceder equation (1) on a squared domain ($\Omega = [0,1] \times [0,1]$), where an exact solution u_{exact} is chosen someway the associated terms g and f of equation (1) are determined such that u_{exact} is the solution of the problem (1) for given coefficient c,a,b and μ that we change.We take for example:

$$u_{exact} = 2sin(2\pi x)cos(2\pi y)$$

We split our domain into four squared sub-domains and we apply the described domain decomposition methods OO2 and AlgDF (one can split the domain to the minimum number of computer unit of execution). We solve equation (2), (3), (6) and (7) by a same finite volume (we use the grunwald scheme [19] to discretize the caputo derivative conditio on the interface in equation (6) and (7)).

To implement our codes in Matlab and Freefeem++ or C++ Language, we work our programs with an intel(R) processor core 2 duo for our computer. We use the parallel package of matlab (parfor, gather, multithread...) and openMP for C++ and we make the discretized(algebraic) equations from problem (1) or (6), (7) on a vectorial and optimal form.
The table 1 show the time that take the OO2 method and the AlgDF to resolve each problem (1) for some given coefficient c, a,b, μ. h is the mesh of the finite volume method.

The table show that the proposed method AlgDF is too fast and optimal in time compared to classical OO2 method For our academic test. The same results is founded for several other academic test cases.

Table 1. On the right CPU time for the OO2 with sex iteration and AlgDF for different coefficient of the equation (1)

Differents coefficients			**OO2 method**	**AlgDF**
c=10 a=1	b=0	$\mu = 1$ $h = 0.0001$	27.621	0.562
c=10 a=-10	b=2	$\mu = 2$ $h = 0.0001$	32.681	0.910
c=10 $a = sin(xy)$	$b = -cos(xy)$	$\mu = 0.1$ $h = 0.0001$	29.713	0.496

The table 2 show the infinite error between the exact solution and the approximate solution using OO2 and AlgDF method. The resulte of the table show that the proposed method is accurate even with one iteration. It take for the OO2 method at least 20 iteration to have a good accuracy.

Table 2. On the right, the infinite norm between the exact solution and the approximated solution by the OO2 with sex iteration and AlgDF for different coefficient of the equation (1)

Differents coefficients			**OO2 method**	**AlgDF**
c=10 a=1	b=0	$\mu = 1$ $h = 0.0001$	$3.4 * 10^{-7}$	$7.8 * 10^{-15}$
c=10 a=-10	b=2	$\mu = 2$ $h = 0.0001$	$4.5 * 10^{-5}$	$3.52 * 10^{-14}$
c=10 $a = sin(xy)$	$b = -cos(xy)$	$\mu = 0.1$ $h = 0.0001$	$1.72 * 10^{-8}$	$6.52 * 10^{-16}$

I notice that the OO2 method take a lot of time to converge for a high viscosity μ (We have studied this case in [1]) but we don't have this issue using our AlgDF. Another important thing to notice is that the OO2 method is sensitive to the accuracy of the mesh eventually when using freefem++ (classical Voronoi and Delanouie mesh) the accuracy of the approximate solution is not efficient.

5 Conclusion

In this work we have developed a new optimized order four DD algorithm applied to a reaction advection diffusion equation. We firstly have computed rate of convergence of this method using the Fourier transform. The fundamental result is that is that our method need one iteration, it doesn't need nor condition of transmission nor optimization time for computing coefficient (like C_1, C_2, C_3) for the transmission condition in comparison with global calculation using classical solvers (OO2, Robin,...). We also proved theoretically and numerically that AlgDF algorithm is more faster than the classical OO2 method. One difficult issue of our method is the calculation of the Fourier transform of non constant and discontinuous coefficients, but in matlab, there is a toolbox to do this job.

Secondly we have presented several test-cases to show the efficiency of this approach.

As perspective of the present work, we can study the following ideas:

- Prove that the method is symmetric. - Perform the method to have one iteration domain decomposition for the Navier Stokes and the Euler compressive equations. We are thinking use some operator decomposition or preeconditionner
- show that the method is dual.
- Perform this method to the equation of Turbulence (The method could be applied by the same way for boundary problem with vector equations) .

References

1. M.R. Amattouch, H. Belhadj, *Combined Optimized Domain Decomposition Method and a Modified Fixed Point Method for Non Linear Diffusion Equation*, Applied Mathematics and Information Sciences, 11, No. 1, 201-207 (2017).

2. M.R. Amattouch, N. Nagid, H. Belhadj, *A modified fixed point method for The Perona Malik equation*,Journal of Mathematics and System Science 7, 175-185, september 2017
3. M.R. Amattouch, N. Nagid, H. Belhadj, *Optimized Domain Decomposition Method for Non Linear Reaction Advection Diffusion Equation*, European Scientific Journal , Vol 12, No 26 (2016).
4. A. SCHWERTNER CHARAO, *Multiprogrammation parallèle générique des méthodes de décomposition de domaine*, Thèse de doctorat, Institut National Polytechnique de Grenoble, France, 2001.
5. S.-L. SOBOLEV, *L'Algorithme de SCHWARZ dans la Théorie de l'élasticité*, Comptes Rendus (Doklady) de l'Académie des Sciences de l'URSS, IV((XIII)6):243-246, 1936.
6. I. BABUSKA , *Uber Schwatzsche Algorithmen in partiallen Differntialgleichungen dermathematischen Physik*, ZAMM, 37(7/8):243-245, 1957.
7. P.-L. LIONS , *On the SCHWARZ alternating method, I. In R. GLOWINSKY, G.-H. GOLUB, G.-A. MEURANT et J. PERIAUX, éditeurs*, First International Symposium on Domain Decomposition Methods for Partial Differential Equations, pages 1-42, Paris, France, 1987. SIAM.
8. P.-L. LIONS , *On the SCHWARZ alternatingmethod. III : a variant for nonoverlapping subdomains. In T. CHAN, R. GLOWINSKY, J. PERIAUX et O. WIDLUND, éditeurs* , Third International Symposium on Domain Decomposition Methods for Partial Differential Equations, pages 202-223, Houston, Texas, USA, 1989. SIAM.
9. A. TOSELLI et O. WIDLUND , *Domain Decomposition Methods - Algorithms and Theory*, volume 34 de Springer Series in ComputationalMathematics. Springer Berlin Heidelberg, 2005. ISBN 3-540-20696-5.
10. Alfio Quarteroni and Alberto Valli, *Domain decomposition methods for partial differential equations*, Oxford Science Publications, 1999.
11. M.J.Gander, *Optimized Schwarz Methods*, SIAM Journal on Numerical Analysis, 44(2), 699-731, 2006.
12. Caroline Japhet, *Méthode de décomposition de domaine et conditions aux limites artificielles en mécanique des fluides : méthode optimisée d'ordre 2* , (Doctoral dissertation, Université Paris-Nord-Paris XIII), 1998.
13. C.A.J. Fletcher *Computational Techniques for Fluid Dynamics 1* Springer 1998.
14. C.A.J. Fletcher *Computational Techniques for Fluid Dynamics 2* Springer 1991.
15. L. Bo ccardo,F. Murat, and J.P. Puel. Résultats d'existence pour certains problèmes elliptiques quasilinéaires. Ann. Scuola Norm. Sup. Pisa Cl. Sci., Serie IV,11(2):213:235,1984.
16. E.A. Co ddington and N. Levinson. Theory of Ordinary Differential Equations. Mac Graw-Hill,New-York,1955.
17. V. Lakshmikantham, S. Leela and J. Vasundhara,Theory of Fractional Dynamic Systems, Cambridge Academic Publishers, Cambridge, 2009.
18. M. Benchohra, J. Henderson, S. K. Ntouyas and A. Ouahab, Existence results for fractional order functional differential equations with infinite delay, J.Math. Anal. Appl. 338(2008), pp.1340-1350.
19. H. Hejazi, T. Moroney, and F. Liu. A finite volume method for solving the two-sided time-space fractional advection-dispersion equation. Central European Journal of Physics, 11(10), 2013.

Planet Wars: an Approach Using Ant Colony Optimization

A. Baldominos[1], M. González-Evstrópova[1], E. Martin[1], and Yago Saez[1]

Computer Science Department, Universidad Carlos III de Madrid, 28911 Leganes (Spain)
Corresponding author: abaldomi@inf.uc3m.es

Abstract. This paper describes an application of Ant Colony Optimization (ACO), a well-known biologically-inspired technique for graph search, for solving a complex multi-objective task posed in the form of a video game, namely Planet Wars, which is itself based on a popular video game known as Galcon and was presented in the Google AI Challenge 2010.
Throughout the paper we introduce the game mechanics and rules and describe how we have applied ACO to solve it, by working on different strategies (expansion, defense and troops rebalancing) and different heuristics which are later submitted to evaluation and whose results are discussed by the end of the paper. These results show that the best performing heuristic is able to beat at least half of the times the Google baseline bots, while in some cases they are defeated in all games and maps.

Keywords. Planet Wars, Video Games, Ant Colony Optimization

1 Introduction

For many years there has been an increasing interest to apply artificial intelligence (AI) techniques to the field of video games. The concept of intelligent non-player characters (NPCs) turns out to be quite promising, and as a result the application of many different AI techniques to video games is explored.

In Fall 2010, Google sponsored the Google AI Challenge [1], which was organized by the University of Waterloo Computer Science Club. In this challenge, contestants had to develop intelligent agents (bots) to win the game Planet Wars. By the end of the challenge, more than 4,600 solutions had been sent by people from 112 different countries.

In this paper, we propose a solution to this problem and challenge using ant colony optimization (ACO), a biologically-inspired AI technique for graph search, where some domain knowledge is introduced in the form of specified heuristics, which tackle different strategies that we have identified as key for solving the game.

The paper is structured as follows: section 2 introduces some related work of recent applications of ACO to the video games domain; section 3 provides the context of the problem by explaining the game mechanics; section 4 describes in a formal manner the solution to the problem using ACO, including the formulation for domain-specific heuristics and the pheromone evaporation and update; section 5 evaluates the proposal by confronting our ACO-based agent to different baseline bots provided by Google; and finally section 6 provides some conclusive remarks and identifies some future lines of work.

2 Related Work

ACO is a technique used for solving combinatorial search problems, and its applications involves many different fields and problems. In a broad sense, only in the last two years applications of ACO include estimating electricity domestic consumption [2], managing water resources [3], rescheduling and rerouting trains [4], planning and controlling robots [5, 6], matching images [7] or evolving deep neural networks [8] to mention a few.

Regarding the field of video games, ACO has also been extensively used. For instance, in 2015 Estrada-Martínez et al. discussed the development of a competitive agent in real-time video games, when fluidity is a critical factor [9]. Chen et al. have studied the problem of integral offensive in the 2D Soccer Simulation League, attaining outstanding performances [10]. Gonzalez-Pardo et al. have compared ACO and GA to solve the *Lemmings* video game [11] and have also provided a formal solution using ACO to CSP-based strategy board games such as *N-Queens* [12].

Not only the development of intelligent NPC agents is studied, as in the work from Połap et al., where the advantages and disadvantages of ACO and BCO (standing for *Bee Colony Optimization*) are presented for the process of constructing boards [13].

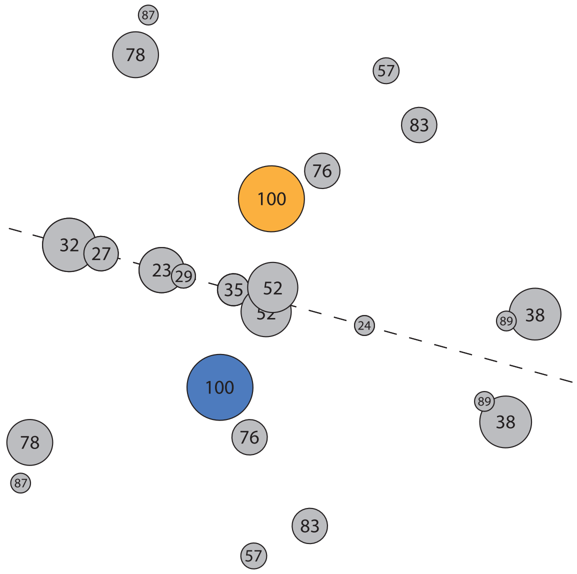


Fig. 1: Sample map for a new game with two players in Planet Wars, showing symmetry around the dashed line.

3 Planet Wars

Planet Wars was a game first introduced in the Google AI Challenge 2010, while being at the time inspired in a popular game available for both iPhone devices and computers known as Galcon and developed by Phil Hassey. The game takes place in a galactic setting, where several players must obtain space troops and conquer planets. A player wins when either he/she has defeated all the enemy troops or has the larger number of troops after the time limit expires.

When the game begins, a map is loaded containing several planets, and following either a central or axial symmetry. Planets are characterized by the next features:

- **Location**: the planet coordinates are used to compute the relative distance among it and its neighbors, measured as the number of game turns required for travelling from one to another.
- **Owner**: the planet may belong to one of the players or to none of them at all. In the latter case, it is designated as a *neutral* planet. When the game starts, all planets are neutral except one for each player.
- **Cost** (C): each planet requires a certain number of troops to be sacrified in order to be conquered.
- **Production rate** (g): planets can produce a certain number of troops in each game turn, as long as it is not a neutral planet.

When troops are sent from one planet to another, they cannot receive further instructions until they have reached their destination. In the case of a confrontation between the players for a planet already owned by one of them, the player with more troops in the planet will win.

Fig. 1 shows an arbitrary map for a new game with two players. Planets are symmetric with respect to the axe depicted with a dashed line. Neutral planets are displayed in light gray along with their cost, whereas the owned planets have a different color for each player.

Players start with 100 troops which must be used to conquer planets, retrieve further troops and defeat the enemy. However this problem remains non-trivial, as players must carefully study their best options in each turn in order to plan a strategy based on the current state of the game to eventually win the game. To simplify the problem, we have defined three srategies which can be played:

- **Expansion**: in the game, attacking the enemy planet in the first turn is not a viable strategy which leads to success. This is due to the fact that when the troops arrive to the enemy planet they will be exceeded in number by the local troops (who, remember, were the same in the beginning of the game but has increased in $t \times g$ individuals). For this reason, the first strategy involves expansion: the agent must decide how to move troops from its own planets to conquer neutral planets.

- **Defense**: at some point in time both players may find themselves fighting for the same planet. In case the enemy wants to take a planet that the agent already owns, the agent must decide whether it makes sense to send troops to that planet, and if so, how many and from which other planets (and more specifically, at what turns).
- **Troops balancing**: after a turn, some troops may remain unused if they have not been sent neither for expansion nor for defense. In that case, the agent might decide to send them elsewhere, so that they can turn useful resources for future strategies.

In many cases, after some turns the agent will have to decide whether to expand to further planets or defend their own ones from the troops of the enemy.

4 Proposal

This section describes the ACO-based system proposed for playing Planet Wars. First, the problem is formalized in terms of graphs and algebra; and later we describe how the agent tackles each of the tasks defined above (expansion, defense and troops balancing).

The problem can be modelled as a graph, where a node represents a planet and a link represents a route between those planets. In fact, this graph will be a complete graph, as all planets are connected among them, i.e. a player can move between every two planets. Let n be the number of planets and $P = \{P_0 \dots P_n\}$ be the set of planets and $P^{(p)} \subseteq P$ the subset of planets owned by player p. The graph can be expressed as a matrix of routes:

$$R = \begin{bmatrix} r_{11} & r_{12} & \dots & r_{1n} \\ r_{21} & r_{22} & \dots & r_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ r_{n1} & r_{n2} & \dots & r_{nn} \end{bmatrix} \tag{1}$$

Every value $r_{ij} \in R$ describes the route between planets i and j, and comprises the following parameters, where D is the route distance, C is the route cost, η is a heuristic value and τ is the amount of pheromone in the route:

$$r_{ij} = \langle D_{ij}, C_j, \eta_{ij}, \tau_{ij} \rangle; \qquad D, C \in \mathbb{Z}; \qquad \eta, \tau \in \mathbb{R} \tag{2}$$

At this point, it should be noticed that an agent cannot send troops from a planet to itself. Also, player p can only traverse routes starting in planets he/she owns. For this reason, the next general constraints are established:

$$r_{ii} = \varnothing; \qquad r_{ij} = \varnothing \text{ if } i \notin P^{(p)} \tag{3}$$

4.1 Expansion

In this case, we want to search for solutions describing which planets must be conquered and in which order, in order to maximize profit. Ants will start this search from the first planet owned by the agent, and as more planets are conquered, ants can also come out from those before deciding where to move. Let Pv be the set of already visited planets and Pc the set of candidate planets for each ant, which is defined as follows:

$$Pc^{(p)} = \{P_i\} \notin P^{(p)} \text{ where } P_i \notin Pv^{(p)} \forall i \tag{4}$$

Each ant will move to a candidate planet according to the next formula, where i is the current planet where the ant is placed:

$$j = \begin{cases} \arg\max\limits_j (\tau_{ij}^{\alpha} \times \eta_{ij}^{\beta}); & \text{if } Q < Q_0 \\ m; \mathbb{P}_{im} = \frac{\tau_{im}^{\alpha} \times \eta_{im}^{\beta}}{\sum_{k=1}^{n} \tau_{ik}^{\alpha} \times \eta_{ik}^{\beta}}; & \text{otherwise} \end{cases} \tag{5}$$

In the previous equation, Q is a random number Q_0 is a defined parameter, where $Q, Q_0 \in [0, 1]$. Parameter Q_0 is determining the relative importance between exploitation and exploration, and the smaller it is, the most likely will be that the ant will move randomly, yet based on a probability distribution which is proportional to the product of the pheromone and heuristic value of each planet.

When estimating the profit of conquering a planet, we must take into account that planets have a growth rate g, but they also have a cost in troops, who are sacrificed during conquest. We have defined four different heuristic functions:

$$\eta_{ij}^{(1)} = \frac{g_j}{D_{ij}^2 \times C_j}; \qquad \eta_{ij}^{(2)} = \frac{1}{C_j}; \qquad \eta_{ij}^{(3)} = \frac{1}{D_{ij}}; \qquad \eta_{ij}^{(4)} = \frac{g_j}{C_j} \tag{6}$$

After ants traverse a route, they release pheromone over it; and some gets evaporated. The pheromone update follows the next equation:

$$\tau_{ij}^{0} = \frac{1}{n \times L_{\min}}; \qquad \tau_{ij}^{t+1} = (1-\sigma)\tau_{ij}^{t} + \sigma\tau_{ij}^{0} \tag{7}$$

In the previous equation, $\sigma \in [0, 1]$ established the pheromone decay parameter and $L_{\min}$ is the shortest path in game turns traversing all the planets.

After an ant moves, the already visited planets are inserted into Pv, and cannot be visited again during the same game turn. When Pc is empty, the ant cannot move towards new planets. Once all ants have finished moving, the solutions they built will be evaluated. To do so, and given that the eventual goal of the expansion phase is to generate troops, we will compare solutions by the number of troops generated by moving to those planets, with $P^{(p)}$ being the set of planets conquered by player p in the evaluated solution; and T and t being the game time limit in turns and the current turn respectively:

$$\mathcal{F} = \sum_{j \in P^{(p)}} ((T - t - D_{ij}) \times g_j - C_j) \tag{8}$$

Once the best solution is chosen, a global update of the pheromone levels take place according to the next equation, where $\varDelta_{ij}$ is the number of troops obtained by the best solution if r_{ij} is part of this solution, or 0 otherwise:

$$\tau_{ij}^{t+1} = (1-\sigma)\tau_{ij}^{t} + \sigma\varDelta_{ij} \tag{9}$$

4.2 Defense

Expansion-only is not a useful strategy, since at some point in time enemy will eventually try to conquer the agent's planets. In practice, this often happens even when there are still neutral planets, as some of the best expansion routes will involve conquering the opponent's planets.

In this case, the goal is no longer finding the most suitable planets for troops generation, but rather finding which of our planets are threatened by the enemy and must be defended. This objective is often opposed to that of the expansion: defending a planet involves sending troops, which could potentially be used for conquering new planets, to them. As a result, a careful decision of whether it is worthy to defend the planet or it is acceptable to lose it in order to gain new ones must be taken.

For the problem formalization, we must include a new constraint over those already observed in equation 3. In particular, this constraint involves being able to send troops only to planets which belong to us, as otherwise would not be a defensive act:

$$r_{ij} = \varnothing \text{ if } j \notin P^{(p)} \tag{10}$$

The ACO process is quite similar to the expansion case, and ants can start the search from any planet owned by the agent; but in this case we want our solution to state which planets to defend and in which order to keep an advantage over the enemy. Also, only planets under attack are considered for the set of candidate planets Pc in a defensive scenario. As a result, equation 4 must be adjusted, with $U^{(p)}$ being the set of troops of player p, U_i the number of troops in planet i and U_{ij} the troops flying from planet i to j:

$$Pc^{(p)} = \{P_i\} \in P^{(p)} \text{ where } P_i \notin Pv^{(p)}, \exists U_{ji} \in U^{(\neg p)} \forall i \tag{11}$$

The transition rule is the same as of equation 5, but the heuristic must be revised for the defense scenario. In this case, the growth rate (g_j) remains an interesting value, as the agent will not want to loose planets which are generating many troops, and so does distance (D_{ij}), as we may not want to defend a planet which is far away, as troops cannot be operated while in-flight. However, cost (C_j) is no longer a parameter to be taken into account, as goal planets are already own by the agent. Also, we will introduce a new parameter, known as the vulnerability of planet j (V_j), defined as follows:

$$V_j = U_j + \sum_{t'=t}^{T} \left(g_j + \sum_i (U_{ij}^{t'}) - \sum_{i'} (U_{i'j}^{t'}) \right); \qquad i \in P^{(p)}, i' \in P^{(\neg p)} \forall i, i' \tag{12}$$

In the previous equation, $U_{ij}^{t'}$ refers to the troops flying from i that will arrive at planet j in time t'. After this definition, we can establish a heuristic:

$$\eta_{ij} = \frac{g_j \times V_j}{1 + D_{ij}} \tag{13}$$

Finally, the fitness function for evaluating different solutions remains as described in equation 8, which makes sense as, after all, defending a planet can be seen as conquering a planet which otherwise would be gained by the enemy. The pheromone evaporation and global update rules also follow equations 7 and 9 respectively; whereas this value is stored separately from the pheromone in the expansion scenario.

4.3 Troops balancing

While expansion and defense are key strategies, we must remember that when troops are sent to one planet, they cannot be operated while on-flight. For this reason, it is interesting to have troops near the *hot-spots*, so that if troops have to be sent to conquer a new planet or defend another, travel times are reduced. Often, these hot-spots will be those planets near the front, i.e. those which are located in the regions between the agent's and the enemy's planets. On the other hand, the rearguard is often a safe place, as it is not likely that the enemy will send troops there and, in case that happens, the agent will often have time to defend the planets.

To formalize this concept, we define the danger index (I_j) for each planet j as the number of enemy's planets which are closer to j than any of the agent's planets, computed as follows:

$$I_j = \sum_i 1 \text{ where } D_{ij} < D_{kj}; \forall i \in P^{(\neg p)}, k \in P^{(p)}$$

The troops balancing phase is rather simple: once the expansion and defense stages are completed, all the generated troops who remain with no destination will be sent to the planet with greater danger index. By doing this, the agent will move troops from the rearguard to the front, where they can play an important role later for further expansion or defense.

5 Evaluation

In this section, we will evaluate our proposal. To do so, we will confront our agents to the bots provided by Google in the starter package. There are five of these bots, which behave as follows:

- **DualBot**: it keeps a constant number of on-flight troops, which are sent from the strongest planet to the weakest enemy or neutral planet. When a certain number of troops arrive, the same number are sent to keep the on-flight troops constant. The planet strength is computed as $S_p = \frac{U_p}{1+g_p}$
- **RageBot**: it only attacks enemy planets and never neutral ones. First, it looks at its own planets p fulfilling $U_p > 10 \times g_p$, and then for each of these planets the closest enemy planet is located and all the troops are sent to that planet.
- **ProspectorBot**: it behaves as DualBot, but there will only be one on-flight troop at any time.
- **BullyBot**: it will attack the enemy's strongest planet, based only of the number of troops in that planet (U_p), with half of the troops it has available.
- **RandomBot**: it moves troops randomly.

For the evaluation, we have arranged 6 games against each of these bots in 20 different maps and for each of the 4 expansion heuristics, thus leading to a total of 2,400 games. The agent will be tested incrementally, by first considering only the expansion strategy, and later considering also the defense strategy, finally taking troop balancing into account.

Figure 2 shows the percentage of won and lost games per each heuristic. Also, as the agent is developed incrementally, the wins are separated by whether the strategy is (a) expansion-only, (b) expansion + defense or (c) expansion + defense + troops rebalancing. It can be seen how expansion by itself is insufficient. The defensive strategy leads to a slight increase in the number of won games, whereas troops rebalancing is critical for significantly improving the agent's performance.

The best heuristics are $\eta^{(1)}$ and $\eta^{(4)}$, i.e. those considering the production rate g, which are able to win 80% and 75% of the games respectively.

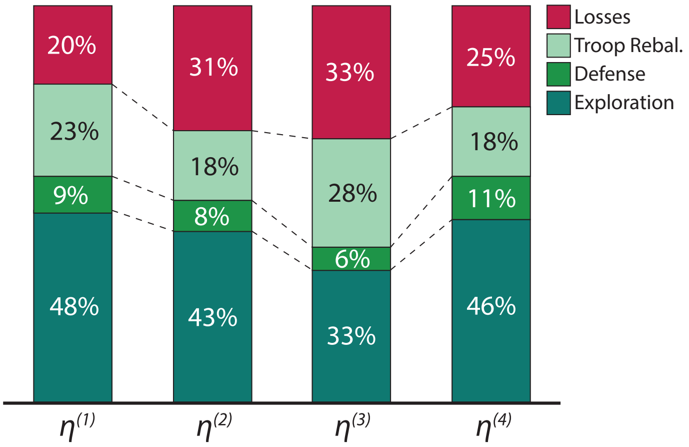
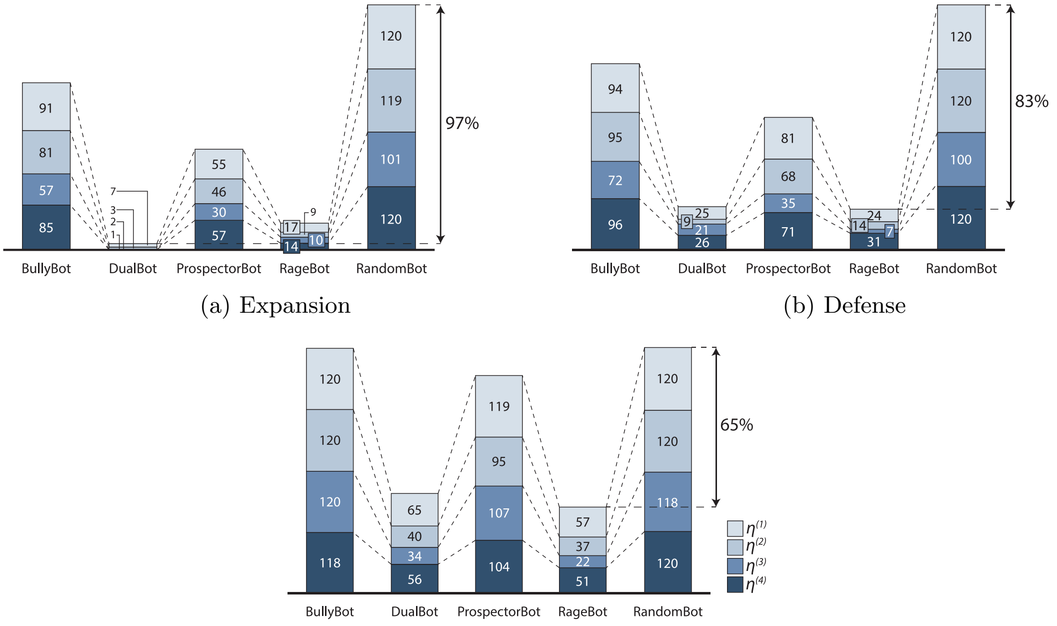


Fig. 2: Percentage of games won/lost for each heuristic, with different strategies.



(a) Expansion

(b) Defense

(c) Troops rebalancing

Fig. 3: Games won for each heuristic and bot type, with different strategies.

Further detail is provided in Fig. 3, which depicts the number of games won per bot and heuristic, from a maximum of 120. In this case it is quite noticeable how adding the defense and troop rebalancing strategies improves the agent's performance, and also decreases the percentage difference between the best and worst performing agents. In all cases, the random bot is easy to beat, but with expansion-only the agent faces real difficulties to beat DualBot and RageBot, which were the bots considering the planets strength or production rate. When all the strategies are observed, the agent with heuristic $\eta^{(1)}$ wins more than half of the games in all cases except for RageBot (57/120). RandomBot, BullyBot and ProspectorBot are easily defeated in all cases.

6 Conclusions and Future Work

Throughout this paper we have explored the application of ant colony optimization (ACO) for solving the Google AI Challenge presented in 2010, which consists in the Planet Wars video game. In this game, players compete for conquering planets and having the larger number of troops.

We have described how ants explore solutions taking into account domain-specific heuristics which follow two major strategies: expansion and defense. Also, we have complemented this solution with a third strategy (not ACO-based) for rebalancing the troops in each game turn. An evaluation shows that heuristics observing the troops production rate and distance provide better results, and when all the strategies are considered the agents are able to win more than half of the games for all Google baseline bots, and in three particular cases (BullyBot, ProspectorBot and RandomBot) agents win all the games. Also, we have proven that expansion by itself in insufficient, and while introducing a defensive behavior improves the agent's performance, troops rebalancing is critical for achieving top results.
However, there is still room for improvement when fighting the more offensive enemies, namely DualBot and RageBot. It is left as future work to explore further defense strategies to increase agent's performance when fighting these bots.

Acknowledgements

This research is supported the Spanish Ministry of Education, Culture and Sport through FPU fellowship with identifier FPU13/03917.

References

1. Google: Google AI Challenge. `http://planetwars.aichallenge.org` (2017) [Online; accessed Oct 12, 2017].
2. Duran-Toksari, M.: A hybrid algorithm of Ant Colony Optimization (ACO) and Iterated Local Search (ILS) for estimating electricity domestic consumption: Case of Turkey. Intl J Electrical Power Energy Syst **78** (2016) 776–782
3. Afshar, A., Massoumi, F., Afshar, A., Mariño, M.: State of the art review of Ant Colony Optimization applications in water resource management. Water Resour Manag **29** (2015) 3891–3904
4. Samà, M., Pellegrini, P., D'Ariano, A., Rodriguez, J., Pacciarelli, D.: Ant colony optimization for the real-time train routing selection problem. Transportaton Res Part B: Method **85** (2016) 89–108
5. Castillo, O., Neyoy, H., Soria, J., Melin, P., Valdez, F.: A new approach for dynamic fuzzy logic parameter tuning in Ant Colony Optimization and its application in fuzzy control of a mobile robot. Appl Soft Comput **28** (2015) 150–159
6. Yao, Y., Ni, Q., Lv, Q., Huang, K.: A novel heterogeneous feature ant colony optimization and its application on robot path planning. In: IEEE Cong Evol Comput (CEC). (2015) 522–528
7. Sreeja, N., Sankar, A.: Ant colony optimization based binary search for efficient point pattern matching in images. Eur J Oper Res **246** (2015) 154–169
8. Desell, T., Clachar, S., Higgins, J., Wild, B.: Evolving Deep Recurrent Neural Networks using Ant Colony Optimization. In: Evolutionary Computation in Combinatorial Optimization. Volume 9026 of LNCS. Springer (2015) 86–98
9. Estrada-Martínez, J., Sánchez-López, A., Jara-Maldonado, M.: On the use of Ant Colony Optimization for video games. In: Advances in Artificial Intelligence and Soft Computing. Volume 9413 of LNCS. Springer (2015) 238–247
10. Chen, S., Lv, G., Wang, X.: Offensive strategy in the 2D Soccer Simulation League using multi-group Ant Colony Optimization. Int J Adv Robotic Syst **13** (2016) 154–169
11. Gonzalez-Pardo, A., Palero, F., Camacho, D.: An empirical study on collective intelligence algorithms for video games problem-solving. Comput Inform **34** (2015) 233–253
12. Gonzalez-Pardo, A., del Ser, J., Camacho, D.: Solving strategy board games using a CSP-based ACO approach. Int J Bio-Inspired Comput (In Press)
13. Połap, D., Woźniak, M., Napoli, C., Tramontana, E.: Is swarm intelligence able to create mazes? Int J Elec Telecomm **61** (2015) 305–310

A Steganographic embedding scheme using Improved-PSO approach

Yamina Mohamed Ben Ali

Computer Science Department, University of Badji Mokhtar,
BP 12, Annaba, 23000, Algeria

Benaliyam2@yahoo.fr

Abstract. In this paper, we are interested by tackling the steganography task as an optimization problem carried out by a bio-inspired approach. Indeed, a novel embedding scheme using the substitution principle of LSB and related to an improved version of particle swarm optimization algorithm is proposed. Therefore, to fulfill the requirement of a good steganographic tool, some updating at the evolution rules of a particle were done. The improved-PSO embedding scheme looks for the best pixels locations, and eventually the best pixels bits to hide secret messages -both text and image- without degrading the quality of the original image. Thus, the robustness of the proposed steganographic scheme relates strongly on the strength of the improved-PSO algorithm in order to achieve higher imperceptibility. Three test cases were undertaken to highlight the performance of the improved-PSO embedding scheme.

Keywords. Adaptive particle swarm optimization, Adaptive inertia weight, Steganography based LSB, PSO embedding scheme.

1. Introduction

The steganography can be considered as a branch of cryptography that tries to hide messages within others [1,2], avoiding the perception that there is some kind of message. Thus, there are two trends to implement steganographic algorithms: the methods that work in the spatial domain, and the methods that work in the transform domain. While the algorithms that work in the transform domain are more robust, that is, more resistant to attacks, the algorithms that work in the spatial domain are simpler and faster.

The most common steganographic technique in spatial domain is the simplest Least Significant Bit Insertion method -LSB- [3,4] in which the hidden message is converted to a stream of bits which replace the pixels values in the cover image. This sort of steganography is only suitable for images stored in bitmap form or losslessly compressed. Unfortunately, it is vulnerable to even a slight image manipulation like converting an image from a format like GIF or BMP [5]. Therefore, many works were done based on both conventional and adaptive scheme of LSB method [6, 7, 8]. However, others researchers looked at others issues such the use of discrete cosine transform -DCT- and others techniques as more powerful steganographic tool [9, 10, 11, 12, 13, 14]. Improving the steganography relies also on the use of bio-inspired algorithms as optimization tool. Indeed, Jackson et al. [15, 16] proposed a computational immune system approach to blind steganography detection. In [17], the authors attempt to use genetic algorithms for achieving the optimal data imperceptibility. Wu et al. in [18] applied the LSB substitution and genetic algorithm to develop two different optimal substitution strategies. A novel steganographic method, based on JPEG and particle swarm optimization algorithm (PSO), is proposed in [19]. In the same trend for achieving more imperceptibility of communicating images, Fazli et al. [20] presented a novel method to embed secret message in the cover-image so that the interceptors will not notice about the existence of the hidden data. In [21] the authors presented a genetic algorithm based method for breaking steganalytic systems. In [22], Wang et al. proposed for more performance to involve both genetic algorithm and a dynamic programming. In [23], the authors applied the ant colony optimization algorithm to construct an optimal LSB substitution matrix. Brazil et al. in [24] proposed a hybrid heuristic, combining a genetic algorithm and the path relinking metaheuristic. Others works have also been reported in [25, 26, 27].

In this paper, a novel improve-PSO embedding scheme is introduce in order to hide a secret message, both text and image, in a cover image with the aim of achieving a higher imperceptibility. The proposed approach uses an improved version of the particle swarm optimization algorithm to carry out the steganography as an optimization problem. Thus, the paper is organized as follows: section 2 describes the used bio-inspired approach PSO, while section 3 describes the proposed approach. In section 4, we provide some experimental tests, and conclude our work by a conclusion.

2. Particle Swarm optimization

PSO is a population-based stochastic optimization technique developed in 1995 [28, 29] inspired by the social behavior of bird flocking or fish schooling. A population of particles randomly initialized in a search space tries to look for the optimal movement. Indeed, each particle i is assigned to it three flying parameters: its current position in the search space $\vec{x}^i$, its current velocity $\vec{v}^i$, in addition to its memorized best past position $\vec{x}^{ipbest}$. However, for the next flying direction at $t+1$, the particles must coordinate with each others to update their positions and velocities relating to the global best particle movement designed by $\vec{x}^{gbe\,st}$. The movement of PSO individuals can be described by Eq. 1 and Eq. 2:

$$\vec{v}^i(\mathrm{t}+1) \ = \omega\vec{v}^i(\mathrm{t}) \ + c_1r_1(\vec{x}^{ipbest} \ - \vec{x}^i(t)) + c_2r_2(\vec{x}^{gbest}\,(t) - \vec{x}^i(t) \qquad (1)$$

$$\vec{x}^i(\mathrm{t}+1) \ = \vec{x}^i(t) + \vec{v}^i(t+1) \qquad (2)$$

The parameter ω is called the inertia factor, $c_1,\ c_2$ are acceleration factors and the random values r_1, r_2 which taking values from $[0,1]$. to best understanding the PSO behavior, let us consider the following algorithm which corresponds to the canonical PSO version. For an n-dimensional problem, the particles are n-dimensional vectors randomly initialized through the search problem space.

3. Optimization Proposed Approach

3.1 Global view

The proposed approach introduces an adaptive embedding scheme based on LSB principle, i.e. hiding a digit of a secret message in one pixel Bit. The objective is then not only to avoid the detection of the hided message but also to avoid that a third person think that such hiding is possible. Figure 1 outlines the principle of the embedding scheme related on the improved PSO algorithm. Thus, each particle at time t represents a stego image (original image incremented with the hidden message) and then must evolve during generations looking for the best stego image that which leads to a more imperceptibility as pertinent criterion. In fact, the encoded parameters of a particle are two vectors of m-dimensions representing respectively the encoded image pixels and bits and where m is the length of the hidden message. The following steps show the main algorithm steps.

- **Step1**: Introduce the hidden message under the text form.
- **Step2**: Convert the text to a binary message. Each symbol in the text is replaced by its ASCII code. At this step, we obtain a binary message constituted by the bytes of symbols.
- **Step3**: Determine the number of pixels needed to hide the binary message.
- **Step4**: Initialize a population of particles with respectively random pixels and random bits positions. Then, hide the “bits of the message in their binary form” in the cover-image at the positions provided the proposed algorithm.
- **Step5**: Evaluate each particle according to the appropriate fitness function (distance) devoted to the steganography purpose.
- **Step6**: Select the best particle that embodies the good pixels (respectively good bits) able to hide the secret message. The best stego-image must be the most similar as the original image with less damage.
- **Step7**: Reiterate the adaptive particle swarm optimization process by updating the particles positions and velocities of pixels and bits until the best stego-image is reached after the termination stopping criterion.

3.2 Improved PSO-embedding scheme

The main improvements reported to the PSO algorithm aim the updating position and velocity rules. Effectively, to accelerate the moves of particles and to provide them the ability to explore far regions of the search space, we have introduced an adaptive inertia weight.

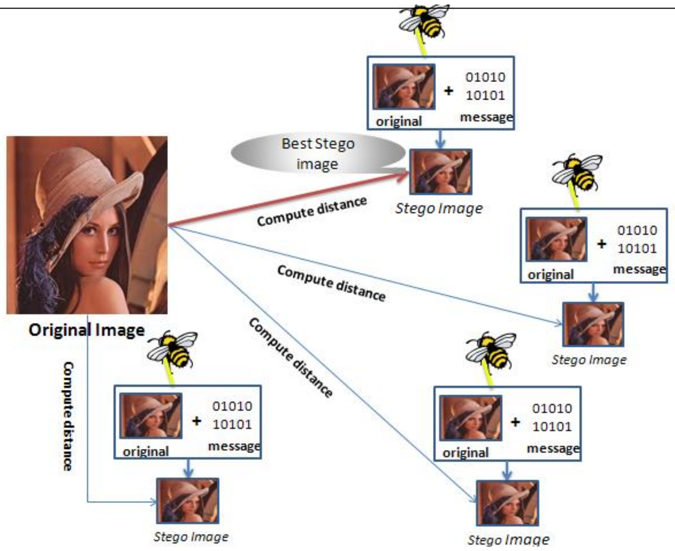


Fig.1 Particle swarm schema to optimize the stego-image

Equation 3 highlights the updating made on this acceleration parameter which promotes at the first generations a higher evolution speed in favor of the exploitation strategy

$$\omega(t+1) = e^{-\omega(t)} \qquad (3)$$

Where $\omega(t=0)$ is set to the empirical value 0.4. However, in order to avoid stagnation in a local optimum especially after some evolution steps, we introduced a second acceleration parameter in Eq. 4 which is devoted to an exploration with large steps jumps. By this mean, we improve considerably the convergence speedup.

$$\eta(t) = \beta * N(0,1) \qquad (4)$$

Where $\beta \in [1,10]$ is a constant that increases in this range as well as the problem size increased. Thus, the new updating rules inspired from Eq. 1 and Eq. 2 were rewritten according to our requirements to highlight respectively the velocity rule (Eq. 5) and the position rule (Eq. 6).

$$\vec{v}^{i}(\mathrm{t}+1) \ = \omega(t)\vec{v}^{i}(\mathrm{t}) \ + c_1 r_1\left(\vec{x}^{t}_{ipbest} \ - \vec{x}^{i}(t)\right) + c_2 r_2(\vec{x}^{gbest}(t) - \vec{x}^{i}(t)) \qquad (5)$$

$$\vec{x}^{i}(\mathrm{t}+1) \ = \vec{x}^{i}(t) + \eta(t)\vec{v}^{i}(t+1) \qquad (6)$$

Obviously, and like any optimization problem, herein a minimization one, the improved embedding scheme requires an objective function that fulfills all prerequisite cited above such as the higher imperceptibility. For this purpose, we used as the mean square error (MSE) measure as the fitness function f by comparing the original and the stego images as illustrated in Eq.7.

$$MSE = \frac{1}{MxN}\sum_{x}\sum_{y}\left(I(x,y) - I^{'}(x,y)\right)^{2} \qquad (7)$$

Where MxN is the size of the image, and $I(x,y)$ and $I^{'}(x,y)$are resp. the intensities of a pixel at the position (x,y) before and after hiding the secret message. Figure 2 detailed the proposed algorithm.

Improved-PSO embedding scheme

```
Step 1.
    •  Initialize the global algorithm parameters: $N = 15; w = 0.7;\ c_1 = c_2 = 2;$
       $v_{min} = -4; v_{max} = 4; nbit = 8;\ dim = size(secret\_message) * nbit, I_{m_{size}};$
    •  Initialize the time counter $t = 0$;
Step 2. Initialize all particles in the swarm population, i.e. $N$ Stego-images $S_m$
    for $i = 1$ to $N$ do
     for $j = 1$to $dim$ do
        $S_m(i,j).Pixel$ = randi ($image_{size}$);
        $S_m(i,j).Bit$ = randi ($nbit$);
        $S(i,j).vel = rand([v_{min}, v_{max}])$;
```

```
          end
        • Keep the best stego-image of particle $i$
          $\vec{S}_m^{ibest} = \vec{S}_m(i,:)$;
      end
  Step 3. Evaluate all the particles fitness $f^i(t)$ according to Eq.7
  Step 4. Select the global best stego image $\vec{S}_m^{gbest}$ at time $t$;
  Repeat
    Step 5. Increment the time counter $t = t + 1$
    Step 6. Find new stego-images according to the improved PSO updating rules
      for $i = 1$ to $N$ do
        for $j = 1$to $dim$ do
          • Updating the inertia parameters $\omega$ and $\eta$ according to resp. Eq.3 and Eq.4
          • Updating the new $S(i,j).vel$ according to Eq.5
          • Updating the new $S_m(i,j).Pixel$ according to Eq.6
          • Updating the new $S_m(i,j).Bit$ according to Eq.6
        end
      end
    Step 7. Evaluate all the particles fitness $f^i(t)$
      for $i = 1$ to $N$ do
        • if $f^i(t) < f^i(t-1)$ then $\vec{S}_m^{ibest} = \vec{S}_m(i,:)$ is the new best local solution
      end
      • Keep the global best stego-image $\vec{S}_m^{gbest}$ at time $t$
    Until Stopping criterion is satisfied.
```

Fig. 2 The improved-PSO embedding scheme

4. Experimental Setup

To evaluate the performance of approach, we have developed an algorithm with Java (programming language) under Eclipse environment using Pentium ® Dual-Core CPU 1.93 GB of RAM. The experiments are divided up in three sub-sections showing each one a performance facet of the proposed embedding scheme.

4.1 Experimental Case1

This experiment aims to compare the performance results between the improved-PSO algorithm and the conventional LSB technique. For this purpose, three cover images illustrated in Figure 3 are considered for the test, in addition to the use of three messages with different sizes $M1$ =303 bytes, $M2$ =342 bytes, and $M3$ = 1078 bytes.

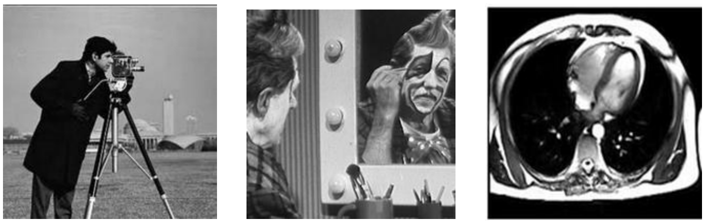


Fig. 3 Three images considered for the first experimental case, from left to right: cameraman, clown, medical.

The performance measures undertaken are the MSE (described above), the Peak Signal-to-Nose Ratio -PSNR- as described by Equation 8, and the Signal-to-Noise Ratio (SNR) described by Equation 9.

$$PSNR = 10\, log_{10} \frac{I_{max}^2}{MSE} \tag{8}$$

Where I_{max} represents the maximal intensity in the image. In the case of grey-level images, I_{max} = 255. The PSNR value approaches infinity as the MSE approaches zero. Thus, a higher PSNR value implies a higher imge quality, and s smaller PSNR value implies a higher differnce between the original image and the best stego image.

$$SNR = 10\, log_{10}\ \frac{I_{av}^2}{MSE} \qquad (9)$$

Where I_{av} represents the average intensity of the target image.

Table 2 shows the achieved comparison results. As we can see, for all tests the improved-PSO algorithm outperforms than LSB. Related to the higher PSNR values (in decibel), the algorithm ensures a higher imperceptibility and excludes the degradation of the cover image. In general, the quality of the stego image is sensitive to the size of the embedded secret message. Indeed, the results show that as well as the message size increases the PSNR and SNR values decrease but still rationally tiny to preserve as even a good performance.

Table 1 Comparison performances between the improved-PSO scheme and LSB.

	Images	Cameraman		Clown		Medical	
		Improved-PSO	LSB	Improved-PSO	LSB	Improved-PSO	LSB
M1	*MSE*	9.46E-3	2.42E-2	9.05E-3	2.43E-2	8.12E-3	2.32E-2
	PSNR	68.3684	64.2801	68.5599	64.2630	69.0342	64.4736
	SNR	62.6701	58.5818	63.0026	58.7057	62.9933	58.4327
M2	*MSE*	1.02E-2	2.73E-2	1.03E-2	2.74E-2	9.37E-3	2.30E-2
	PSNR	68.0191	63.7658	67.9887	63.7467	68.4127	64.4961
	SNR	62.3208	58.0674	62.4314	58.1894	62.3719	58.4553
M3	*MSE*	3.37E-2	8.11E-2	3.32E-2	8.09E-2	3.03E-2	5.11E-2
	PSNR	62.8435	59.0368	62.9151	59.0471	63.3093	61.0436
	SNR	57.1452	53.3385	57.3578	53.4897	57.2685	55.0028

4.2 Experimental Case2

In this experiment, we intend to compare the improved-PSO scheme with two efficient steganography methods, the algorithm described in [17] and JQTM described in [24] based upon PSNR performances. For this purpose, six grayscale images secret images of Lena, Mandrill, Woman, Boat, Goldhill and Girl with size 256x256 are considered. The size of the secret image in [24] is 104x64, and those of [17] is 128x72. For testing our algorithm, we choose to take as message size the higher, i.e. also 128x72. Figure 4 shows three selected results achieved by our algorithm. Visually, we can observe that compared to the original images, the best stego images seem similar and show no degradation at the colors level. Unfortunately, although the worst stego images remain visually understandable, the distinguishable colors degradation suggests that some anomalies like hiding a message inside is strongly recommended.

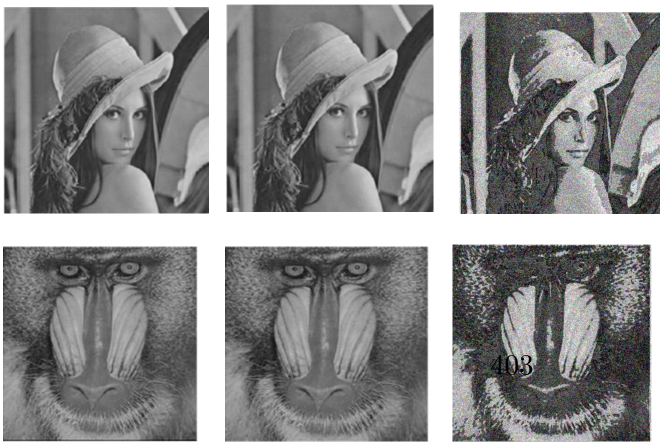
Table 2 The image quality (PSNR) of JQTM, [17], and the proposed scheme.

Methods	Lena	Mandrill	Woman	Boat	Goldhill	Girl
JQTM	36.81	31.17	35.42	36.05	36.43	37.64
[17]	37.06	31.28	35.71	36.29	36.78	38.02
Improved-PSO	41.81	39.15	41.41	41.05	41.44	42.56

Table 3 illustrates the achieved results provided by the improved-PSO scheme and those reported by their authors in respectively [24] and [17]. It is clear that not only the improved-PSO scheme outperforms as algorithm compared to its competitors but provides acceptable PSNR values in the range of the invisible hiding.

4.3 Experimental Case3

This experiment emphasis on the fact that not only the imperceptibility is a crucial criterion to reach also preserving the quality of the stego image against some attacks when communicated it is a second criterion to aim.



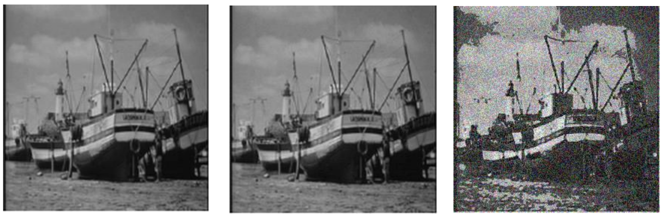


Fig. 3 From left to right : Original image, best stego-image, and worst stego-image of respectively Lena, Mandrill, and Boat images.

Indeed, three JPEG cover images of resolution 512x512 in addition to a secret image of size 64x64 are used. Thus, to measure the robustness of the used approach, we take into account the normalized correlation (NC) coefficient described by Equation 10.

$$NC(X,Y) = \frac{\sum_{i=1}^{L} \sum_{j=1}^{L} [X(i,j) * (Y(i,j)}{\sum_{i=1}^{L} \sum_{j=1}^{L} [X^2(i,j)} \qquad (10)$$

where L is the size of embedded image, and X and Y are respectively the embedded image (after hiding) and its extraction after undergoing attacks. As cited in literature, conventional attacks in image processing further to geometric attacks are performed. The first category gathers Gaussian noise with zero mean and 0.01 variance, salt and pepper noise, Poisson noises are added to the stego image, in addition to mean and median filters of size 3x3 and 5x5. The second category gathers cropping attacks with different rates 10%, 20%, and 40%, scaling attacks with 0.8, 1.2 and 1.5 ratios, the 1- degree rotation attack, histogram equalization, and the compression attack. Obviously, the results shown in Tables 4-5 are reported from their respective references [24] and [25].

As we can see on Table 4, and in comparison with the results of reference [25], the proposed algorithm outperforms and exhibits real improvement especially with regard to the noise generated by the mean filter of size 3x3 and 5x5. This phenomenon is due to the fact that the noise is dispersed in the entire image without any distinction between specific zones. On the other hand, against geometric attacks, Table 5 highlights less successful results in respect of the proposed approach and compared to the results of [24]. Indeed, the cropping attack decreases relatively the performance of the proposed method since whenever the attacked zone increases the normalized correlation value decreases in consequence until reaching a NC value of 0.4781 on Mandrill image. The same behavior is also observed for the scalex0.8 attack. Thus, for these two kind of attacks, the results reported in [24] are better compared to ours. This phenomenon is due to the fact that the attacked zone is compact and the pixels of the same neighborhood have undergone to serious intensities modifications. For the rest of attacks, the proposed algorithm shows best results than those of reference [24].

Table 4 Robustness performances against conventional image processing attacks (NC values).

Image	**Algorithm**	**Noise addition**			
		Gaussian	**Salt &Pepper**	**Poisson**	
Lena	Improved-PSO	1.0019	0.9964	1.0003	
	[25]	0.9204	0.9149	0.9562	
Mandrill	Improved-PSO	0.9905	0.9980	0.9990	
	[25]	0.8511	0.8975	0.9387	
Peppers	Improved-PSO	0.9764	0.9888	0.9963	
	[25]	0.9196	0.9010	0.9493	
Image	**Algorithm**	**Median 3x3**	**Median 5x5**	**Mean 3x3**	**Mean 5x5**
Lena	Improved-PSO	0.9991	0.9982	0.9986	0.9968
	[25]	0.9785	0.9029	0.6577	0.6577
Mandrill	Improved-PSO	0.9826	0.9754	0.9798	0.9714
	[25]	0.8806	0.8867	0.6730	0.6722
Peppers	Improved-PSO	0.9991	0.9977	1.0051	1.0075
	[25]	0.9599	0.9053	0.6419	0.6419

Table 5 Robustness performances against conventional geometric attacks (NC values).

Image	Algorithm	Crop 10%	Crop 20%	Crop 40%	Hist. Equalization
Lena	Improved-PSO	0.9056	0.8450	0.6193	1.0227
	[24]	0.9378	0.8861	0.7803	0.9503
Mandrill	Improved-PSO	0.8461	0.7102	0.4781	1.0495
	[24]	0.9243	0.8757	0.8204	0.7936
Peppers	Improved-PSO	0.8898	0.7890	0.6060	0.9864
	[24]	0.9283	0.8722	0.7654	0.9698
Image	Algorithm	**Scale x0.8**	**Scale x1.2**	**Scale x1.5**	**Rotation 1-degree**
Lena	Improved-PSO	0.7586	0.9575	0.9483	0.9825
	[24]	0.9478	0.9388	0.9046	0.8757
Mandrill	Improved-PSO	0.7490	0.9364	0.9060	0.9592
	[24]	0.8608	0.8586	0.8144	0.7881
Peppers	Improved-PSO	0.7827	0.9715	0.9519	0.9842
	[24]	0.9599	0.9633	0.8933	0.8593

5. Conclusion

This paper presents a steganographic approach as an improved-PSO embedding scheme which uses the substitution principle to embed a secret message inside an image. Thus, the approach consists at looking for the best pixels and best bits in order to maximize the imperceptibility and then to minimize the error between the original cover image and the stego to avoid discrimination between them. The proposed approach easy to implement has shown good results with regard to the size of the embedding secret message. Other performances which consist in adding image processing attacks have also been carried out in order to evaluate the robustness of the approach. Provided the achieved results, we can say that the reported results in comparison with other algorithms in literature highlight more performance.

References

[1] Chandramouli, R., Kharrazi, M., Memon, N.: Image Steganography and Steganalysis: Concepts and Practice. In: LNCS 2939, 35-49 (2004).

[2] Donovan, Artz: Digital steganography: hiding data within data. In: IEEE Internet Computing 5(3), 75-80 (2001).

[3] Fridrich, Jessica, Goljan, Miroslav, Du Rui: Reliable Detection of LSB Steganography in Color and Grayscale Images. In: Magazine of IEEE Multimedia, Special Issue on Multimedia and Security, 8, 22-28 (2001).

[4] Ker, Andrew D.: Improved detection of LSB steganography in grayscale images. In: Proc. of 6th Information Hiding Workshop, LNCS 3200, 97-115 (2004).

[5] Chandramouli, R., Memon, N.: Adaptive of LSB based image steganography techniques. In: Proc. of International Conference on Image Processing 3, 1019-1022 (2001).

[6] Manoharan, S.: Towards robust steganography using T-codes. In Proc. 4th EURASIP Conference focused on Video/Image Processing and Multimedia Communications, 707-711 (2003).

[7] Chan, Chi-Kwong, Cheng, L.M.: Hiding data in images by simple LSB substitution. Pattern Recognition 37(3), 469-474 (2004).

[8] Sakakura, T., Hayashi, A.: A simple detection scheme of LSB steganography based on statistics of image difference signal. In: Proc. of the International Symposium on Information Theory and its Applications, 320-325 (2010).

[9] Tseng, Hsien-Wen, Chang, Chin-Chen: High capacity data hiding in JPEG-compressed images. Informatica 15, 127-142 (2004).

[10] Muttoo, S.K., Kumar, Sushil: Secure image steganography based on slantlet transform. In: Proc. of International Conference on Methods and Models in Computer Science, 102-108 (2009).

[11] Chin-Chen Chang and Hsien-Wen Tseng: Data hiding in images by hybrid LSB substitution", Proc. of the Third International Conference on Multimedia Ubiquitous Engineering, 360-363 (2009).

[12] Huang, Der-Chen, Chan, Yung-Kuan, Wu, Jhin-Han: An agent-based LSB substitution image hiding method. International Journal of Innovative Computing, Information and Control 6(3), 1023-1038 (2010).

[13] Rabevohitra, F.H., Sang, Jun: Using PSO Algorithm for Simple LSB Substitution Based Steganography Scheme in DCT Transformation Domain. In: Advances in Swarm Intelligence, LNCS 6728, 212-220 (2011).

[14] Chang, C.C., Chen, T.S., Chung, L.Z.: A steganographic method based upon JPEG and quantization table modification. Information Science 141, 123–138, (2002).

[15] Jackson, J.T., Gunsch, G. H., Claypoole, R.L., Lamont G.B.: Blind steganography detection using a computational immune system: a work in progress. International Journal of Digital Evidence 4(1), 1-19 (2003).

[16] Jackson J.T., Gunsch, G.H., Claypoole, R.L., Lamont, G.B.: Novel steganography detection using an artificial immune system approach. In: Proc. of the Congress on Evolutionary Computation, 139-145 (2003).

[17] Maity, Santi P., Kundu, Malay K., Nandi, Prasanta K.: Genetic algorithm for optimal imperceptibility in image communication through noisy channel. In: Proc. of the International Conference on Neural Information Processing, LNCS 3316, 700-705 (2004).

[18] Wu, Ming-Ni, Li, Min Hui, Chang, C.C.: A LSB substitution oriented image hiding strategy using genetic algorithms. In: Proc. of the Advanced Workshop On Content Computing, LNCS 3309, pp. 219-229 (2004).

[19] Li, Xiaoxia, Wang, Jianjun: A steganographic method based upon JPEG and particle swarm optimization algorithm. Information Sciences 177, 3099-3109 (2007).

[20] Fazli, S., Kiamini, M.: A high_performance steganographic method using JPEG and PSO algorithm. In: Proc. of the IEEE International Multitopic Conference, 100-105 (2008).

[21] Shih, Frank Y., Wu, Yi-Ta: Digital steganography based on genetic algorithm. In: Handbook of Research on Secure Multimedia Distribution, 439-453 (2009).

[22] Yang, Cheng-Hsing, Wang, Shiuh-Jeng: Transforming LSB substitution for image-based steganography in matching algorithms. Journal of Information Science and Engineering 26(4), 1199-1212 (2010).

[23] Hsu, Ching-Sheng, Tu, Shu-Fen: Finding optimal LSB substitution using ant colony optimization algorithm. In: Proc. of the Second International Conference on Communication Software and Networks, 293-297 (2010).

[24] Brazil, Andre Luiz, Sanchez, Angel, Conci, Aura, Behlilovic, Narcis: Hybridizing genetic algorithms and path relinking for steganography. In Proc. ELMAR, 285-288 (2011).

[25] Vahedi, Ehsan, Zoroofi, Reza Aghaeizadeh, Shiva, Mohsen: Toward a new wavelet-based watermarking approach for color images using bio-inspired optimization principles. Digital Signal Processing 22, 153-162 (2012).

[26] Fakhari, Pegah, Vahedi, Ehsan, Lucas, Caro: Protecting patient privacy from unauthorized release of medical images using a bio-inspired wavelet-based watermarking approach. Digital Signal Processing 21, 433-446 (2011).

[27] Al-Qaheri, Hameed: Digital watermarking using ant colony optimization in fractional Fourier domain. Journal of Information Hiding and Multimedia Signal Processing 1(3), 179-189, (2010).

[28] Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proc. of the IEEE International Conference on Neural Networks, 1942-1948 (1995).

[29] Kennedy, J., Eberhart, R.C.: Swarm intelligence. Morgan Kaufmann, (2001).

Backpropagation and PSO-GA based Optimizations for a Neural Network Classification of Engine Fault Signals

S. Mjahed[1(*)], S. El Hadaj[1], K. Bouzaachane[1], S. Raghay[1]

[1)] Department of Applied Mathematics and Computer Sciences
Faculty of Sciences and Technology
Cadi Ayyad University
40000 Marrakech, Morocco

(*) Corresponding author: soukaina.mjahed@gmail.com

Keywords: Classification, Backpropagation (BP), Neural Network (NN), Particle Swarm Optimization (PSO), Genetic Algorithm (GA), Fault Signals

Abstract

The fault diagnosis is important to ensure normal operation of an engine. An effective way to diagnose engine fault is to schedule maintenance operations by adopting the concept of condition-based maintenance (CBM). It consists on a predictive maintenance that helps to make cost saving decisions by observing the equipment's health and act only when maintenance is necessary. This technique reduces the spent effort and the economic losses and leads to make wise decisions with a reduced error probability and minimal environmental damages [1]-[2].

There are two ways to diagnose faulty engines: supervised learning classification and unsupervised learning classification called clustering. Classification has been widely used in several domains, especially engine condition-based maintenance [2].

Many classification algorithms have been proposed for equipment health prediction [3]. Different techniques for fault detection and isolation in process of engine monitoring are suggested such as Neural Networks [4]-[6], Particle Swarm Optimization technique [7]-[8], Genetic Algorithms [9] and Support Vector Machine [10].

In this work, we attempt to apply an improved classification algorithm based on Backpropagation Neural Network for solving multi-objective detection and diagnosis of fault signals in industrial processes using real data. Robust stochastic optimization techniques like Particle Swarm Optimization (PSO) and Genetic Algorithm (GA) are compared with Backpropagation (BP).

We mainly focus on classifying some machines sound signals data into 4 classes corresponding to a normal class and three anomalous gear sound signals classes. We consider three types of engine faults: faulty bearings, joints problem, and mechanical loosening. We model the data using six discriminative sound signals features in order to classify them into the four classes defined above with higher performances.

The performances of the proposed approaches (BP Neural Network, PSO and GA based Neural Networks) are evaluated by using the efficiency and the purity of classifications.

The results demonstrate the effectiveness of the PSO and GA based Neural Networks compared to Backpropagation Neural Network. Our approaches compare favorably with other machine learning techniques.

References

[1] Li-Juan, G., Chun-hui Z., Min H. and Yong Z. (2013). Vibration analysis of the steam turbine shafting caused by steam flow. *Telkomnika, 11(8): 4422-4432.*

[2] W. J. Moore and A. G. Starr (2006). An intelligent maintenance system for continuous cost-based prioritisation of maintenance activities, *Computers in industry, 57: 595–606.*

[3] S. K. Padhi, A. Mahapatra, P.Mishra (2013). Multi-fault classification using various soft computing techniques: a survey, *International Journal of Engineering Research & Technology, 2(9): 1908-1911.*

[4] Y. Kourd, D. Lefebvre, N. Guersi (2013). Fault diagnosis based on neural networks and decision trees: application to damadics. *International Journal Of Innovative Computing, Information And Control, 9(8): 3185-3196.*

[5] Abhijit Suresha, K.V Harisha, N. Radhikaa (2015). Particle swarm optimization over back propagation neural network for length of stay prediction, *Procedia Computer Science 46: 268–275.*

[6] Jiri Krenek, Kamil Kuca, Pavel Blazek, Ondrej Krejcar and Daniel Jun (2016). Application of Artificial Neural Networks in Condition Based Predictive Maintenance, *Studies in Computational Intelligence 642: 75–86.*

[7] P. Rini, M. Shamsuddin, S. Yuhaniz (2011). Particle Swarm Optimization: technique, system and challenges, *International Journal of Computer Applications 14: 0019–0027.*

[8] Dong-Han Lee, Jong-Hyo Ahn, and Bong-Hwan Koh (2017). Fault Detection of Bearing Systems through EEMD and Optimization Algorithm, *Sensors (Basel). 17(11): 2477.*

[9] M. Compare, F. Martini, E. Zio (2016). Genetic algorithms for condition-based maintenance optimization under uncertainty, *European Journal of Operational Research, 244(2): 611-623.*

[10] V. Sugumaran, and K.I. Ramachandran (2011). Effect of number of features on classification of roller bearing faults using SVM and PSVM, *Expert Systems with Applications, 38: 4088-4096.*

Customer Order Scheduling by Scattered Wolf Packs

Vahid Riahi[1], M A Hakim Newton[1], M M A Polash[2], and Abdul Sattar[1]

[1] Institute of Integrated and Intelligence Systems (IIIS), Griffith University, Australia
vahid.riahi@griffithuni.edu.au,mahakim.newton@griffith.edu.au,a.sttar@griffith.edu.au
[2] Computer Science and Engineering, Jagannath University, Bangladesh polash@cse.jnu.ac.bd

Abstract. Customer Order Scheduling Problem (COSP) is NP-Hard. COSP has important practical applications e.g. in the paper industry and the pharmaceutical industry. In this paper, we empirically advance the state of the art of COSP by outperforming a very recently presented Greedy Search Algorithm and by finding new upper bounds for 728 out of 960 benchmark instances. We achieve these results by using a new scattered wolf pack (SWP) algorithm that works on discrete combinatorial optimisation problems. SWP hybridises various elements of scatter search and grey wolf optimisation. SWP encompasses wolf pack characteristics such as leadership hierarchy and hunting mechanisms that are mostly intensification oriented. SWP further encompasses plausible wandering behaviours and scattering mechanisms that are essential for search diversification.

1 Introduction

A customer order scheduling problem (COSP) [1] has n customer orders and m parallel machines. Each customer order j is composed of m items. Each item i is processed by only a particular machine i. Machines are able to process different items of a single customer order j at the same time. Each item i of a customer order j needs a known, deterministic and a non-negative period of processing time $p_{ij} \geq 0$ at machine i. The completion time C_j of a customer order j is the time point when processing of all items of the customer order is finished. The aim for COSP is to find a sequence of the customer orders at each machine such that the summation of the completion times of the customer orders is minimised. There are as many possible sequences of customer orders as the permutations on all machines and the number is $(n!)^m$. However, the optimal solution is found when the permutation of the customer orders is the same for all machines [2]. Therefore, the search space of COSP with n customer orders comprises $n!$ permutations.

COSP has several real-life applications. In a car repair shop [3], there are several mechanics and each arriving car has several broken parts. Each broken part requires service from a particular mechanic and mechanics can work on the same car at the same time. A car leaves the repair shop only when all of its broken parts are fixed. This car repair shop model can also be adapted to aircraft maintenance and ship repair [3]. Moreover, COSP is also relevant in the pharmaceutical industry [4], in the manufacturing of semi finished lenses [5], and in the paper industry [6].

COSP is an NP-Hard problem [6] meaning it is difficult to be optimally solved especially when the problem is large. Therefore, researchers recently put much more attention on developing approximate and metaheuristic COSP algorithms. The current state-of-the-art results are obtained by a very recent Greedy Search Algorithm (GSA) [7]. In this paper, we empirically advance the state of the art of COSP by outperforming GSA and by finding new upper bounds for 728 out of 960 benchmark instances. We achieve these results by using a newly proposed Scattered Wolf Pack (SWP) algorithm that works on discrete combinatorial optimisation problems.

SWP hybridises various elements of Scatter Search (SS) [8] and Grey Wolf Optimisation (GWO) [9]. SWP encompasses wolf pack characteristics such as leadership hierarchy and hunting mechanisms that are mostly intensification oriented. SWP further encompasses plausible wandering behaviours and scattering mechanisms that are essential for search diversification. Technically, like GWO, to intensify the search, we combine three best solutions to generate new solutions. Other solutions in the population are like the diverse population in SS. We combine these other solutions and the generated solutions. We use a multi-neighbourhood local search to escape local optima. We further use an explicit diversity control mechanism.

In the rest of the paper, the COSP model and related work are discussed in Section 2, SS and GWO are briefly covered in Section 3, the proposed SWP algorithm is described in Section 4, computation results are provided in Section 5, and conclusions are presented in Section 6.

2 Customer Order Scheduling Problems

Assume $\pi \equiv \pi_n$ is a permutation of n customer orders $1, 2, \ldots, n$ and $[k]$ denotes the kth customer order in π_n where $k \in [1, n]$. Also, assume $C_{i[k]}$ is the completion time point of customer order $[k]$ at machine i. Given the processing time $p_{i[k]}$ for a customer order $[k]$ at machine i, $C_{i[k]} = C_{i[k-1]} + p_{i[k]}$ when $k > 1$ and $C_{i[1]} = p_{i[1]}$. For a customer order $[k]$, the completion time point is $C_{[k]} = \max_{i=1}^{n} C_{i[k]}$. For a permutation $\pi \equiv \pi_n$, the total completion time is $C(\pi) = C(\pi_n) = \sum_{i=k}^{n} C_{[k]}$. Fig. 1 depicts a COSP with 3 machines and 4 customer orders. $C_{[1]}$, $C_{[2]}$, $C_{[3]}$, and $C_{[4]}$ are the customer order completion times and the total completion time is $C(\pi) = C_{[1]} + C_{[2]} + C_{[3]} + C_{[4]}$.

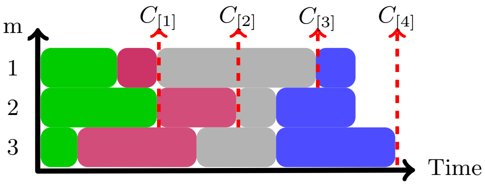


Fig. 1. A COSP with three machines and four orders.

Various heuristic methods have been presented for COSP. *Shortest Total Processing Time (STPT)* heuristic [10] starts with an empty sequence. Then it constructs a schedule by putting the unscheduled customer order with the smallest total processing times into the last spot of the partial sequence. STPT is very fast but the performance significantly decreases especially with the increasing of the problem size since it only focuses on the last spot of the partial solution. *Earliest Completion Time (ECT)* [6] inserts all unscheduled customer orders one by one at the last position of the partial sequence, and the lowest one in terms of the objective function is chosen. This process continues until all customer orders are scheduled. SPT-B [7]. FP heuristic [7] (named abbreviating the authors' names) starts with an initial sequence obtained by SPT-B. Next, similar to ECT, customer orders are tested on the last spot of the partial sequence. However, FP always considers a complete sequence considering those unscheduled customer orders based on the STP-B ordering as the rest of the partial sequence. Focusing only again on the last spot leads to performance degradation for both FP and ECT especially with increasing size of the problem.

Among metaheuristic methods, a Tabu Search (TS) [6] and a Greedy Search Algorithm (GSA) [7] are also proposed for COST. TS used ECT for generating an initial solution and then used the swap operator in local search. GSA employed FP for the initial solution. Then it goes through the main loop which contains three steps. At the first step, one customer order is randomly selected and moved to the last spot. At the second step, GSA finds the best neighbour for the customer order newly occupied the last spot, and finally applies an exhaustive swap operator on the solution obtained from the second step. It is continued until the stopping criterion is met. Experimental results showed the efficiency of GSA over the TS algorithm. However, both algorithms exhaustively used all possible swap moves; which increases the required CPU times as well as the risk of getting stuck in the local optima.

3 Scatter Search and Grey Wolf Optimisation

Scatter Search: SS [8] is an evolutionary metaheuristic for integer programming. SS creates two reference set of solutions of similar sizes: one containing the best solutions found so far and the other containing diverse solutions. In each iteration, SS then intelligently combines solutions from the two reference sets to generate new solutions that encompass both quality and diversity. Each generated solution then undergo a local improvement method. The reference sets are updated at the end of each iteration. The main characteristic of SS is the diversification of the solutions as a means to high quality optimisation.

Grey Wolf Optimisation: GWO is a recently emerged population-based metaheuristic for continuous optimisation problems. GWO borrows its ideas from basic living and hunting behaviours of grey wolves [9]. Wolf packs exhibit a leadership hierarchy: Alpha, Beta, and Delta wolves are the three highest level leaders in the order, while the rest of the wolves in the pack are Omega wolves. Lower level wolves always follow higher level wolves in performing activities that include hunting, sleeping, group protection, and territorial inspection. GWO starts with a random initial population of wolves. Each wolf represents a solution of the problem. In each iteration, the three best solutions are considered to be the Alpha, Beta, and Delta wolves in the order and GWO computes each Omega solution combining itself with the Alpha, Beta, and Delta solutions.

4 Proposed Scattered Wolf Pack Algorithm for COSP

As mentioned before, our proposed SWP population-based algorithm combines various elements of SS and GWO. We borrow three ideas from SS: keeping diverse solutions in the population, combining good solutions with diverse solutions, and applying local search algorithm on generated solutions. From GWO, we borrow the idea of keeping only three best solutions, and combining these three best solutions with each diverse solution. Moreover, we use a multi-neighbourhood local search to escape local optima. We further use an explicit diversity control mechanism.

Algorithm 1 shows the pseudocode of the the SWP algorithm. Below we describe each component of the SWP algorithm in details. However, before that we give an overview. After initialisation of the population of a given size N, the three best solutions α, β, and δ (in terms of the objective function) in the population undergo local search. In each iteration of the algorithm, each of the other solutions in the population gets replaced by a new solution. The new solution is found by applying a combination method name SB3OX on the three best solutions and then applying a path-relinking procedure. With some probability, local search is also performed on the new solution. In each iteration, a certain percentage (P_{LS}) of the worst solutions in the population are then replaced by randomly generated solutions. The three best solutions then again undergo local search. When the search terminates, the best solution is returned.

```
Algorithm 1: The proposed SWP algorithm
Parameters: population size $N$, diversity controller proportion $P_{\text{DC}}\%$, local search probability $P_{\text{LS}}$
Population Initialisation: Generate one solution using a given heuristic and the others randomly.
Best Solution Selection: Three best solutions in the population are $\alpha$, $\beta$, and $\delta$ respectively.
Perform Local Search: $\pi \leftarrow$ performLocalSearch($\pi$) where $\pi \in \{\alpha, \beta, \delta\}$.
while termination condition is not met do
    for each solution $\omega$ in the population except $\alpha$, $\beta$, $\delta$ do
        Update Solution: $\omega \leftarrow$ performPathRelinking(performSB3OX($\alpha, \beta, \delta$), $\omega$).
        Perform Local Search: $\omega \leftarrow$ performLocalSearch($\pi$) with probability $P_{\text{LS}}$.
    Diversity Control Mechanism: Replace $P_{\text{DC}}\%$ of worst solutions with new random ones.
    Best Solution Selection: Three best solutions in the population are $\alpha$, $\beta$, and $\delta$ respectively.
    Perform Local Search: $\pi \leftarrow$ performLocalSearch($\pi$) where $\pi \in \{\alpha, \beta, \delta\}$.
Output: The best solution found.
```

4.1 Popolation Initialization

Initial solutions could be typically generated randomly. However, using heuristic algorithms can improve the quality of the solutions and obtain an initial population with a high level diversity [11]. SWP uses *Earliest Completion Time* (ECT) heuristic [6] to generate one solution, and the remaining solutions are generated randomly. ECT starts with an empty sequence π, and all unscheduled costumer orders are placed in a list $\mathcal{L}$. At each iteration $1 \leq k \leq n$, ECT separately inserts each unscheduled customer order to the kth spot (the last spot) of the partial sequence π. The customer order that leads to the lowest objective function is picked for position k, and is then removed from list $\mathcal{L}$. This process continues until $\mathcal{L}$ is empty.

4.2 Perform Local Search

As can be seen in Algorithm 1, we always apply local search on each of the three best solutions in the population. Also, we apply local search on the other solutions in the population, but with some probability; this is to avoid too much exploration of the inferior solutions. A careful observation of the algorithm might also reveal that for some best solutions, local search is applied on the same solution more than once. This is not a bad aspect since our local search does not explore the neighbourhood of a given solution exhaustively. Each time the local search is applied to the same solution, it will probabilistically explore new areas around the solution.

Algorithm 2 shows the local search algorithm that we use in SWP for COSP. In each iteration, with 50%-50% probabilities, we apply either insert or swap moves. Insert and swap moves are typically used when solutions can be represented by permutations. In the insert move, a random customer order $[k']$ is removed from its position k' in the permutation and then reinserted at a random position $k'' \neq k'$. In the swap move, two random customer orders $[k'] \neq [k'']$ are selected from the solution and their positions are exchanged. The parameter $20n$ in the termination criteria is set after some preliminary experiments and denote the maximum allowed plateau length.

Algorithm 2: performLocalSearch

```
Let $\pi$ is the input solution, and $l \leftarrow 1$.
while $++l \leq 20n$ do
    $\pi' \leftarrow$ insert($\pi$) or swap($\pi$)
        with probability 50%.
    if $C(\pi') < C(\pi)$ then
        $\pi \leftarrow \pi'$, $l \leftarrow 1$
return $\pi$
```

Since calculation of the objective function from scratch for each new solution would be time-consuming, in this work, we present a fast neighbourhood evaluation strategy called *acceleration method*. The *acceleration method* is simple but effective and is similar to the one in [12]. The idea is to compute only the completion time of the changed part of the permutation after applying each move and reuse the unchanged part of the permutation. Suppose permutation π be the current solution. First, a matrix size $n \times m$ is created that retains the completion times of each customer order on each machine. If k is the minimum of k' and k'' used in the insert or swap moves, then the completion times of the first k customer orders can be easily reused in computation of the new solutions after applying the move.

4.3 Update Solutions

For each solution ω in the population except α, β, and δ, in Algorithm 1, we first combine the three best solutions α, β, and δ by using a method named SB3OX to generate a new solution ω'. We then apply path-relinking on ω' and ω to get a solution that replaces ω in the population. The motivation to use the path-relinking procedure is to mimic the behaviour that an ω wolf is moving from its current position to the position obtained by combining the three best solutions. The path-relinking procedure also helps obtain diversity since the SB3OX for each ω in the same iteration will create a new solution always from the same α, β, and δ.

SB3OX. This method is based on a crossover operator named Similar Block 2-Point Order Crossover(SB2OX) [13]. SB2OX showed its effectiveness over other typical crossover operators used in genetic algorithms and other population based metaheuristics. In the original SB2OX operator, it creates two child solutions from two parent solutions. In this paper, we modify this crossover so that it creates one solution from three solutions α, β, δ and name it as Similar Block 3-Point Order Crossover(SB3OX). In SB3OX, first all positions of the three best solutions are checked. The blocks of at least two consecutive customer orders that have exactly the same position in all three solutions are directly copied to the new solution. Next, two cut points are randomly chosen to divide the solutions in three parts. Then, the first part of the new omega solution is filled in from the first part of α, the second part is filled in by the second part of β, and the third part is filled in by the third part of δ. In the filling in process, any duplicates in the new solution are replaced by unused customer orders in the other parts of the source solutions.

Path-Relinking Procedure. Path-relinking [14] is originally used in the tabu search to incorporate intensification and diversification strategies. Path-relinking creates a route from a given source solution s to a given target solution t by applying a sequence of moves. Starting from the source s, at each step a given type of move is applied on the current solution on the route to obtain the next solution on the route. The intermediate solutions on the route thus have varying levels of similarities with the source and destination solutions. In this paper, we use swap moves to move from s to t. To show few steps using an example, consider $s = (2, 6, 3, 1, 5, 4)$ and $t = (3, 5, 6, 2, 4, 1)$. Swapping customer orders 2 and 3 in s will produce $s' = (3, 6, 2, 1, 5, 4)$, which has the same customer order 3 at the first position as t has. Next swapping customer orders 5 and 6 in s' will produce $s'' = (3, 5, 2, 1, 6, 4)$, which have the same first two customer orders as t has.

4.4 Diversity Controlling Mechanism

Since three best solutions are used to generate the other solutions in the population, diversity of the generated solutions are still an issue. In order to tackle that, we use a diversity control mechanism [15, 16]. As part of the mechanism, we replace the worst $P_{\mathrm{DC}}\%$ solutions from the population with randomly generated solutions. However, maintaining a certain level of distance between a newly generated solution and the solution being replaced should be considered in order to take the search to a different area of the search space. For this, in this work, we use two distance-based measures, which are described below. For convenience, assume $[k]_\pi$ and $[k]_{\pi'}$ denote the customer orders at the kth positions of the solutions π and π' respectively.

The first distance measure is D the most common one found in the literature in various problems. It is the number of positions where the two solutions have different customer orders scheduled.

$$D(\pi, \pi') = \textstyle\sum_{k=1}^{n} d([k]_\pi, [k]_{\pi'}) \text{ where } d(j, j') = 1 \text{ if } j \neq j' \text{ else } 0$$

Unfortunately, this measure is not much effective. For example, when $\pi = (1, 2, 3, 4, 5, 6)$ and $\pi' = (6, 1, 2, 3, 4, 5)$, D is the highest distance 6. However, if we see carefully, only one relocation of 6 at the beginning of π will result in π'. So the distance between π and π' should perhaps be 1 instead. To address this issue to some extent, the second distance measure we use is D' [11]. This distance measure takes into account pairs of customer orders at successive positions in either direction. If $[k]$ and $[k+1]$ in π neither match respectively with any $[k']$ and $[k'+1]$ in π' nor do so with any $[k'+1]$ and $[k']$ in π', then a mismatch is counted. According to D', the distance between $\pi = (1, 2, 3, 4, 5, 6)$ and $\pi' = (6, 1, 2, 3, 4, 5)$ is 1 since only $(5, 6)$ pair in π is not in π'.

$$D'(\pi, \pi') = \textstyle\sum_{k=1}^{n-1} d'([k]_\pi, [k+1]_\pi, \pi') \text{ where}$$
$$d'(j, j', \pi') = 0 \text{ if } \exists_{k'}(([k']_{\pi'} = j \wedge [k'+1]_{\pi'} = j') \vee ([k'+1]_{\pi'} = j \wedge [k']_{\pi'} = j')) \text{ else } 1$$

Given the definition of D and D', for each candidate worse solution ω to be replaced by a new solution ω', we enforce $D(\omega, \omega') = n$ and $D'(\omega, \omega') = n - 1$. These are the maximum distances possible for the measures respectively and indicate high diversity levels.

5 Computational results

We use the benchmark set generated by [7] to evaluate our algorithm. This benchmark is made up of 1680 instances: 720 small and 960 big instances. In this paper, only the 960 big instances are used. The 960 big instances are categorised into two subgroups, Test-1-B and Test-2-B, each containing 480 instances. The difference between these two groups is that in the former, each customer order comprises m items with $p_{ij} > 0$, while in the latter, $p_{ij} \geq 0$ [7]. In further details, both these subgroups contain 16 $n \times m$ combinations where $n \in \{20, 50, 100, 200\}$ and $m \in \{2, 5, 10, 20\}$. Thus, each combination includes 30 instances making 480 in total.

The best known metaheuristic for COSP is GSA [7]. Therefore, in this paper we compare the proposed SWP algorithm with GSA. Because of space restriction, we leave the comparison of the SWP approach against the original SS and GWO based algorithms for an extended report. Both SWP and GSA algorithms are implemented in C programming language. The stopping criterion for both the algorithms were the maximum elapsed CPU time limit of $T_{max} = \rho \times n \times m$ milliseconds, where ρ was tested with three values: 90, 180 and 270. The algorithms were run on the same computing cluster named Gowonda at Griffith University, Australia. Each node of the cluster was equipped with Intel Xeon CPU E5-2670 processors @2.60 GHz and FDR 4x InfiniBand Interconnect, having a system peak performance of 18,949.2 Gflops. All statistical tests were carried out with minitab 18.1. To compare the performances of the algorithms, we use the relative percentage deviation RPD $= \dfrac{(C_{\pi'} - C_{\pi})}{C_{\pi}} \times 100$, where $C_{\pi'}$ is total completion time of the solution return by a given algorithm, and C_{π} is the reference total completion time for the problem instance. The smaller the RPD value, the better the algorithm performance. For convenience of comparison, we also take average of RPD (called ARPD) over the runs of the same instance, over all the instances in a given combination or even in all combinations.

5.1 Parameter Tuning

Calibration of parameters has a great impact on the efficiency of stochastic algorithms. The proposed SWP has three parameters: N the size of population, P_{LS} local search probability, and P_{DC} diversity controller percentage. The following levels are considered for each parameter: $N \in \{10, 20, 30\}$, $P_{\text{LS}} \in \{0.05, 0.15, 0.25\}$, and $P_{\text{DC}} \in \{10, 20, 30, 40\}$ resulting in to $3 \times 3 \times 4 = 36$ variants. To avoid biased results, the benchmark problem set used for parameter calibration is different from that used for algorithm comparisons. Here, we generated 60 instances with $n \in \{20, 50, 100, 200\}$ and $m \in \{5, 10, 20\}$, 12 $n \times m$ combinations and 5 instances each. Each instance in each combination was executed 5 times. The maximum elapsed CPU time was set as $T_{max} = 270 \times n \times m$ milliseconds. In RPD calculation, the reference C_{π} is the best solution obtained by all 36 parameter variants.

Table 1. ANOVA test results for SWP parameter calibration.

Source	DF	Adj SS	Adj MS	F-Value	P-Value
P_LS	2	0.04016	0.02008	374.79	**0.000**
P_DC	3	0.17551	0.058504	1091.98	**0.000**
N	2	0.68764	0.34382	6417.38	**0.000**
P_LS*P_DC	6	0.00006	0.00001	0.19	0.979
P_LS*N	4	0.00024	0.000061	1.14	0.336
P_DC*N	6	0.00053	0.000089	1.66	0.126
Error	10776	0.57734	0.000054		
Total	10799	1.48149			

To analyse the results of the experiment, the Analysis of Variance (ANOVA) method is used. The ANOVA table is shown in Table 1. As can be seen, all factors have high F-ratios and are statistically significant as p-values are close to zero. Furthermore, N (size of population) has the highest F-ratio which means that it has a great effect on the proposed algorithm. We also show the ARPDs and the 95% confidence intervals with Tukeys Honest Significant Difference (HSD) of the factors in Figure 2. Note that non-overlapping confidence intervals of each pair indicates a significant difference. Based on the Figure, we fix N to 20, P_{LS} to 0.15, and P_{DC} to 20 for our further experiments.

5.2 Effectiveness of SWP Components

The proposed SWP algorithm has three important components: a path relinking (PR) technique, a diversity controller (DC) mechanism, and local search algorithm (LS). To test the effectiveness

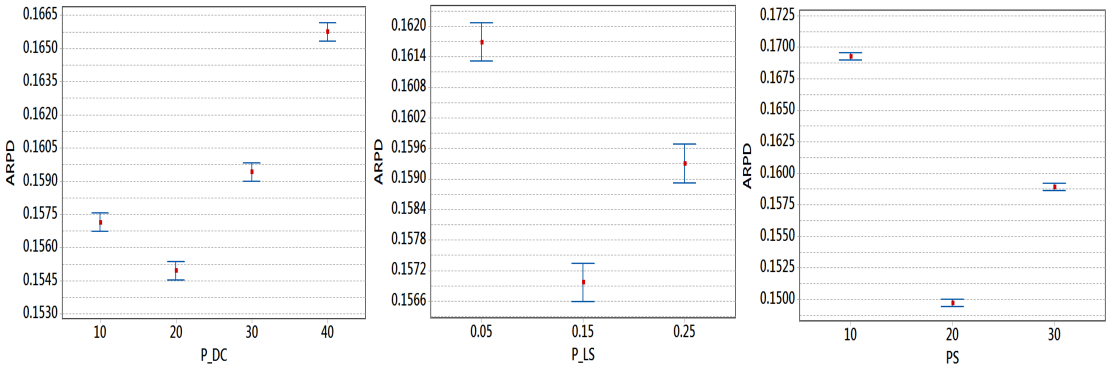


Fig. 2. ARPDs and 95% confidence intervals with Tukeys Honest Significant Difference (HSD) for different levels of parameters.

of these components, we create several variants of the SWP algorithm. One is SWP itself that has all the components; another is +PR which has only the PR component but not the other two; yet another is −LS which does not have LS but has PR and DC component, and the last one is NIL which has none of three components. In this experiment, the same stopping criterion of $270nm$ milliseconds is used for all four variants. Also, the benchmark problems instances are the 60 generated problem instances explained in Section 5.1. All algorithm variants are executed 5 times and the ARPDs are calculated using the best solution found by these variants as the reference. The ANOVA table and Means plot with Tukey's Honest Significant Difference (HSD) 95% confidence intervals are given in Table 2 and Figure 3 respectively. From these results, we can find that each of the components improve the efficiency of the proposed SWP, the path relining procedure making the most improvement.

Table 2. ANOVA test results for SWP variants.

Source	DF	Adj SS	Adj MS	F-Value	P-Value
Algorithms	3	165.9	55.3026	367.39	**0.000**
Error	1196	180.0	0.1505		
Total	1199	345.9			

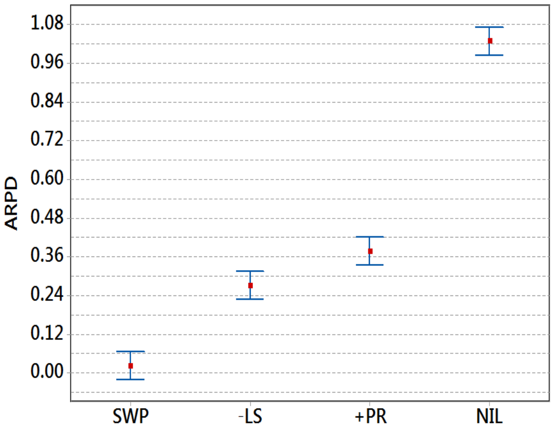


Fig. 3. ARPDs and 95% confidence intervals with Tukeys Honest Significant Difference (HSD) for different variants of SWP.

5.3 Effectiveness of the proposed SWP algorithm

Finally, we compare the proposed SWP algorithm against GSA [7] the best performing existing algorithm designed for the same problem. The ARPD calculation uses the best known solutions found in the literature and also found in this paper. Each algorithm over five runs are reported in Tables 3 and 4 for TEST-1-B and TEST-2-B respectively.

Table 3. ARPDs of the algorithms for TEST-1-B. Best values in each timeout is in bold.

TEST-1-B	$\rho = 90$		$\rho = 180$		$\rho = 270$	
n×m	SWP	GSA	SWP	GSA	SWP	GSA
20×2	**0.000**	0.150	**0.000**	0.147	**0.000**	0.144
20×5	**0.000**	0.369	**0.000**	0.369	**0.000**	0.376
20×10	**0.000**	0.634	**0.000**	0.608	**0.000**	0.613
20×20	**0.000**	0.686	**0.000**	0.693	**0.000**	0.687
50×20	**0.000**	0.133	**0.000**	0.136	**0.000**	0.144
50×5	**0.007**	0.604	**0.004**	0.612	**0.003**	0.620
50×10	**0.029**	0.990	**0.018**	1.003	**0.015**	1.001
50×20	**0.037**	1.048	**0.041**	1.049	**0.031**	1.038
100×20	**0.005**	0.157	**0.003**	0.120	**0.003**	0.118
100×5	**0.054**	0.577	**0.040**	0.557	**0.021**	0.561
100×10	**0.080**	1.075	**0.049**	1.076	**0.038**	1.067
100×20	**0.096**	1.334	**0.075**	1.318	**0.068**	1.353
200×2	**0.007**	0.570	**0.004**	0.359	**0.003**	0.256
200×5	**0.061**	0.801	**0.036**	0.635	**0.027**	0.552
200×10	**0.110**	1.211	**0.070**	1.083	**0.036**	1.027
200×20	**0.119**	1.450	**0.079**	1.422	**0.060**	1.387
Average	**0.038**	0.737	**0.026**	0.699	**0.019**	0.684

Table 4. ARPDs of the algorithms for TEST-2-B. Best values in each timeout is in bold.

TEST-2-B	$\rho = 90$		$\rho = 180$		$\rho = 270$	
n×m	SWP	GSA	SWP	GSA	SWP	GSA
20×2	**0.000**	0.115	**0.000**	0.121	**0.000**	0.122
20×5	**0.000**	0.433	**0.000**	0.424	**0.000**	0.411
20×10	**0.000**	0.738	**0.000**	0.712	**0.000**	0.771
20×20	**0.000**	0.552	**0.000**	0.535	**0.000**	0.549
50×20	**0.000**	0.140	**0.000**	0.138	**0.000**	0.145
50×5	**0.005**	0.649	**0.003**	0.669	**0.002**	0.666
50×10	**0.010**	0.999	**0.004**	1.031	**0.005**	1.010
50×20	**0.008**	1.311	**0.007**	1.300	**0.005**	1.308
100×20	**0.003**	0.172	**0.002**	0.134	**0.002**	0.129
100×5	**0.060**	0.736	**0.040**	0.674	**0.025**	0.682
100×10	**0.081**	1.211	**0.062**	1.155	**0.047**	1.141
100×20	**0.087**	1.548	**0.069**	1.530	**0.057**	1.529
200×2	**0.005**	0.563	**0.004**	0.366	**0.003**	0.310
200×5	**0.069**	1.548	**0.048**	1.158	**0.032**	0.936
200×10	**0.120**	1.915	**0.077**	1.557	**0.064**	1.369
200×20	**0.154**	2.306	**0.094**	2.057	**0.085**	1.987
Average	**0.038**	0.933	**0.026**	0.848	**0.020**	0.817

From Tables 3 and 4, we can see that the proposed SWP algorithm is much better than GSA with all stopping criterion. To be sure that these results are statistically significant, first an ANOVA test is done considering the two algorithms as the factors. The ANOVA results in Table 5 shows that there is a significant difference between algorithm as p-value was 0.000 for both testbeds. The 95% confidence interval plot of the algorithms based on ARPDs of each of the 480 instances is shown in Figure 4 to find out the significance of difference between the pair of algorithms. From this figure we can see that the differences between proposed SWP and GSA is statistically significant.

To see the behaviour of the algorithms for various numbers of customer orders and machines, the interactions between m and n with the algorithms are shown in Figure 5. As can be seen

Table 5. ANOVA test results for compared algorithms on TEST-1-B and TEST-2-B.

	Source	DF	Adj SS	Adj MS	F-Value	P-Value
TEST-1-B	Algorithm	5	1664	332.729	2529.48	**0.000**
	Error	14394	1893	0132		
	Total	14399	3557			
TEST-2-B	Algorithm	5	2546	509.172	2022.47	**0.000**
	Error	14394	3624	0.252		
	Total	14399	6170			

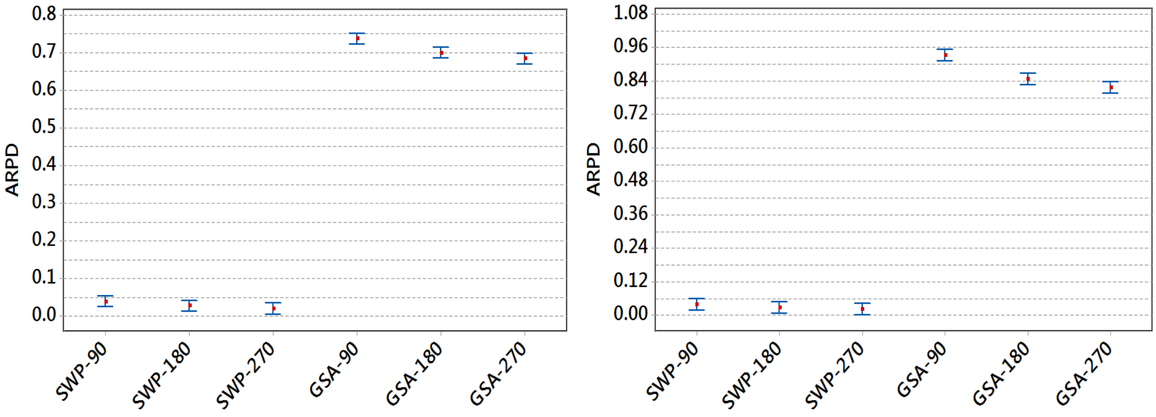


Fig. 4. Average RPD and 95% confidence intervals with Tukeys Honest Significant Difference (HSD) for compared algorithms.

from this figure, for both the numbers of customer orders and machines, the performance of the proposed SWP algorithm is less affected, but GSA performs poorly in both cases especially when the numbers of machines increases.

Finally, it must be mentioned that out of 960 instances, new best solutions are found for 728 instances by the proposed SWP. To be more particular, out of 240 instances with size $n = 200$ i.e. in the largest problems, the proposed SWP finds new best solutions for 239 instances. Note that due to space restrictions, the new best know solutions are not shown here in detail, but we will report these in an extended version of the paper.

6 Conclusions

In this paper, we study the Customer Order Scheduling Problem (COSP) that has realistic applications such as in the paper industry and the pharmaceutical industry. The COSP is known to be NP-hard. In this paper, we empirically advance the state of the art of COSP by outperforming a very recently presented Greedy Search Algorithm and by finding new upper bounds for 728 out of 960 benchmark instances. For this we have developed a new scattered wolf pack (SWP) algorithm that works on discrete combinatorial optimisation problems. The proposed SWP hybridises various elements of scatter search (SS) and grey wolf optimisation (GWO). SWP encompasses wolf pack characteristics such as leadership hierarchy and hunting mechanisms that are mostly intensification oriented. SWP further encompasses plausible wandering behaviours and scattering mechanisms that are essential for search diversification. Technically, like GWO, to intensify the search, we combine three best solutions to generate new solutions. Other solutions in the population are like the diverse population in SS. We combine these other solutions and the generated solutions. We use a multi-neighbourhood local search to escape local optima. We further use an explicit diversity control mechanism. Each distinct SWP component significantly improves the performance. In future, we intent to perform an extended study of the algorithm particularly in terms of the comparison with original SS and GWO algorithms.

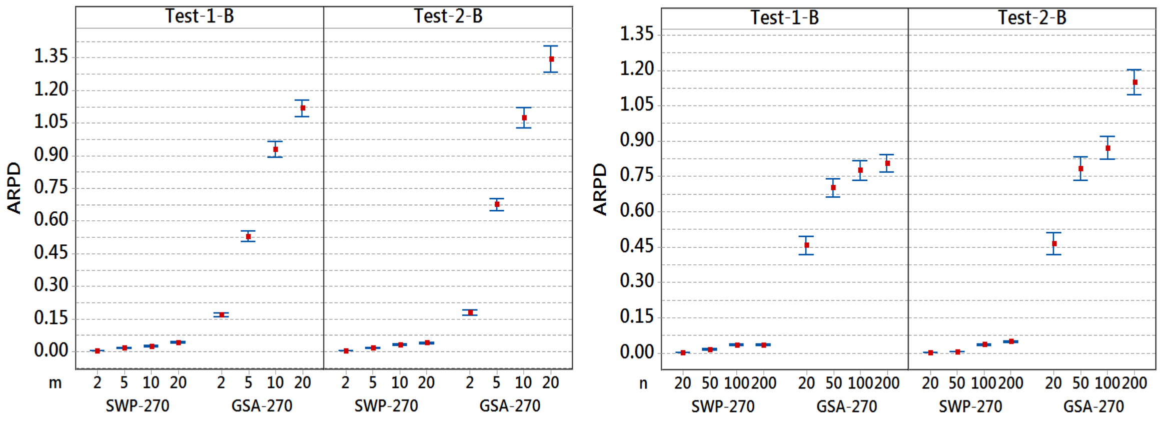


Fig. 5. 95% confidence interval of the competing algorithms on various numbers of machines (left) and customer orders (right).

References

1. Ahmadi, R., Bagchi, U.: Scheduling of multi-job customer orders in multi-machine environments. ORSA/TIMS, Philadelphia (1990)
2. Lee, I.S.: Minimizing total tardiness for the order scheduling problem. International Journal of Production Economics **144** (2013) 128–134
3. Yang, J.: Scheduling with batch objectives. PhD thesis, The Ohio State University (1998)
4. Leung, J., Li, H., Pinedo, M.: Multidisciplinary scheduling: Theory and applications. Chapter Order Scheduling Models: an overview (2005)
5. Ahmadi, R., Bagchi, U., Roemer, T.A.: Coordinated scheduling of customer orders for quick response. Naval Research Logistics (NRL) **52** (2005) 493–512
6. Leung, J.Y.T., Li, H., Pinedo, M.: Order scheduling in an environment with dedicated resources in parallel. Journal of Scheduling **8** (2005) 355–386
7. Framinan, J.M., Perez-Gonzalez, P.: New approximate algorithms for the customer order scheduling problem with total completion time objective. Computers & Operations Research **78** (2017) 181–192
8. Glover, F.: Heuristics for integer programming using surrogate constraints. Decision sciences **8** (1977) 156–166
9. Mirjalili, S., Mirjalili, S.M., Lewis, A.: Grey wolf optimizer. Advances in engineering software **69** (2014) 46–61
10. Sung, C.S., Yoon, S.H.: Minimizing total weighted completion time at a pre-assembly stage composed of two feeding machines. International Journal of Production Economics **54** (1998) 247–255
11. Riahi, V., Khorramizadeh, M., Newton, M.H., Sattar, A.: Scatter search for mixed blocking flowshop scheduling. Expert Systems with Applications **79** (2017) 20–32
12. Li, X., Wang, Q., Wu, C.: Efficient composite heuristics for total flowtime minimization in permutation flow shops. Omega **37** (2009) 155–164
13. Ruiz, R., Maroto, C., Alcaraz, J.: Solving the flowshop scheduling problem with sequence dependent setup times using advanced metaheuristics. European Journal of Operational Research **165** (2005) 34–54
14. Glover, F., Laguna, M.: Tabu Search. Kluwer Academic Publishers (1997)
15. Pan, Q.K., Ruiz, R.: An estimation of distribution algorithm for lot-streaming flow shop problems with setup times. Omega **40** (2012) 166–180
16. Ruiz, R., Maroto, C., Alcaraz, J.: Two new robust genetic algorithms for the flowshop scheduling problem. Omega **34** (2006) 461–476

For solving the multi-depot fleet size and mix open vehicle routing problem

S. Ismail[1], R. Lahyani[2,3], and H. Chabchoub[3,4]

[1] MODILS Laboratory, FSEG, Sfax, Tunisia
ismail.sabrine88@yahoo.com
[2] College of Busines, Alfaisal University, Riyadh, KSA
[3] LOGIQ Laboratory, ISGI, Sfax, Tunisia
rlahyani@alfaisal.edu
[4] School of Business, Al Ain University of Science and Technology, UAE
habib.chabchoub@aau.ac.ae

1 Introduction

This paper introduces a new variant of the vehicle routing problem named the multi-depot fleet size and mix open vehicle routing problem (MDFSMOVRP). In the MDFSMOVRP, the vehicle fleet is composed of vehicles having different capacities and different fixed and variable costs starting from different depots and ending ate the last customer served. An example of this problem appears when a transportation company having different depots hires different vehicles to perform some deliveries. The objective of the MDFSMOVRP is to minimize the routing costs. Although the problem has a wide applicability in real-life applications, it has not been studied in any research work so far, to the best of our knowledge. Some works in the literature studied some reductions of the MDFSMOVRP such as the multi-depot fleet size and mix vehicle routing problem (MDFSMVRP), the single-depot fleet size and mix open vehicle routing problem (FSMOVRP), and the multi-depot open vehicle routing problem (MDOVRP) with homogeneous fleet. These three variants will be briefly reviewed in what follows.

Few attempts have been made recently to solve the closest variant to the MDFSMOVRP which is the MDFSMVRP. In 2014 Vidal et al. [14] proposed a unified dynamic programming methodology to solve several vehicle routing problem variants including the MDFSMVRP. The authors presented a multi-start Iterated Local Search (ILS) and an Hybrid Genetic Search with Advanced Diversity Control (HGSADC). Salhi et al. [11] proposed a mixed integer linear program and a variable neighborhood search algorithm to solve the problem. Recently, in 2018, Lahyani et al. [4] proposed five distinct formulations to model the MDFSMVRP and a branch-and-cut and a branch-and-bound algorithms to solve it. Other works solved extended versions of the MDFSMVRP, e.g., the MDFSMVRP with time windows in [2] and the pickup and delivery MDVRP with heterogeneous fleet in [3].

In what concerns the FSMOVRP, Ren [9] proposed an efficient genetic algorithm using a sequence of a real numbers coding. Yousefikhoshbakht et al. [16] developed a bone route algorithm followed by a tabu search algorithm as a local search procedure. In 2015, the authors proposed a mixed integer program along with an ant colony algorithm and generated a new set of instances to test the problem in [17]. Recently, Yousefikhoshbakht et al. [18] proposed a combined heuristic algorithm called SISEC based on column generation (CG). Compared to the exact and CG algorithms, the proposed algorithm is better in terms of running time and solution quality.

Many exact and approximate methods have been proposed to study the MDOVRP. The problem was firstly studied by Tarantilis and Kiranoudis [13] to solve a real-life distribution problem through a list-based threshold accepting algorithm. Liu et al. [6] proposed a mixed integer program and a hybrid genetic algorithm while [8] developed an integer linear programming model and a branch-and-cut bound algorithm to solve small size MDVRP's benchmarks along with a simulated annealing algorithm to solve medium and large-sized benchmarks. Yao et al. [15] proposed an ant colony optimization to solve the MDOVRP occuring in a seafood delivery application encountered in Dalian in China. Later on, Lalla-Ruiz et al. [5] proposed a mixed integer program to model the MDOVRP and propose some new lifting constraints. Soto et al. [12] presented a general multiple neighborhood search hybridized with tabu search algorithm.

The remainder of this paper is organized as follows. Section 2 provides a description of the problem studied. A mathematical formulation to model the problem is presented in Section 3. In Section 4 we propose an hybridized Adaptive Large Neighborhood search (ALNS) to solve the MDFSMOVRP. Computational results on small and large classes of benchmark instances are reported in Section 5. Section 6 is devoted for conclusions.

2 Problem description

The MDFSMOVRP can be defined on a direct graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$ where $\mathcal{V}$ is the vertex set and $\mathcal{A}$ is the arc set. The set of arcs $\mathcal{A}$ is composed of $\{(i, j) : i, j \in \mathcal{V}, i \neq j\}$. The set $\mathcal{V} = \mathcal{V}_d \cup \mathcal{V}_c$ includes the subset $\mathcal{V}_d = \{1, \ldots, m\}$ that represents m depots whereas the subset $\mathcal{V}_c = \{m+1, \ldots, m+n\}$ represents n customers. A demand q_i is associated with each customer $i \in \mathcal{V}_c$, while $q_i = 0\ \forall i \in \mathcal{V}_d$. The travel distance from node i to node $j \in \mathcal{V}$ is represented by β_{ij}. Loop arcs, (i,i), between customers $i \in \mathcal{V}_c$ and between depots $i \in \mathcal{V}_d$ are forbidden. This is imposed by defining $\beta_{ij} = \infty$ $\forall i \in \mathcal{V}$ and $\beta_{ij} = \infty\ \forall i, j \in \mathcal{V}_d$. $\mathcal{G}$ is a complete graph as it includes all arcs connecting distinct pairs of vertices, with the exception of loops. An heterogeneous fleet of vehicles $\mathcal{K} = \{1, ..., K\}$ having different capacities start from different depots $d \in \mathcal{V}_d$ and complete their mission on the last customer. The capacity of all vehicles of type $k \in \mathcal{K}$ is denoted by Q^k. A fixed cost F^k and a variable cost α^k per unit of distance are associated to each vehicle type.

A solution to the problem must determine routes that minimize the total costs such that each route must originates at one of the depots and terminates at one of the customers, each customer is visited exactly once, and the total demand of each route does not exceed the capacity of the selected vehicle. In addition, the vehicle fleet composition will be determined for each depot. An illustration of the MDFSMOVRP solution with 2 depots, 10 customers and 3 vehicle types is shown in Figure 1.

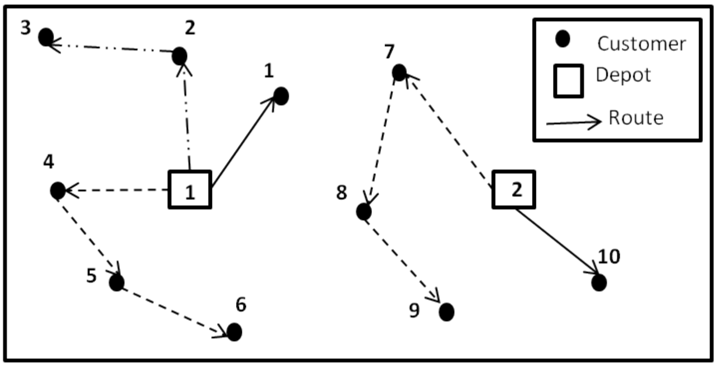


Fig. 1. An example of a MDFSMOVRP solution

3 Mathematical formulation

In this section, we present a mixed integer linear program to model the MDFSMOVRP. This formulation is derived from the commodity flow model proposed inLahyani et al. [4] for the MDFSMVRP. This formulation is based on routing variables and loading variables. Let x_{ij}^{kd} be binary routing variables that take value one if arc $(i, j) \in \mathcal{A}$ is traversed by vehicle type k housed at depot d and zero otherwise. Binary variables y_i^{kd} are equal to one if customer i is visited by vehicle k that starts from depot d and zero otherwise. Loading continuous variables z_{ij}^{k} are used to determine the vehicle type on each arc. The MDFSMOVRP can be formulated as follows:

$$\text{minimize Z} = \sum_{i \in \mathcal{V}_c} \sum_{k \in \mathcal{K}} \sum_{d \in \mathcal{V}_d} F^k x_{di}^{kd} + \sum_{(i,j) \in \mathcal{A}} \sum_{k \in \mathcal{K}} \sum_{d \in \mathcal{V}_d} \alpha^k \beta_{ij} x_{ij}^{kd} \tag{1}$$

subject to

$$\sum_{k\in\mathcal{K}}\sum_{d\in\mathcal{V}_d} y_i^{kd} = 1 \quad i\in\mathcal{V}_c \tag{2}$$

$$y_i^{kd} \leq y_d^{kd} \quad i\in\mathcal{V}_c, k\in\mathcal{K}, d\in\mathcal{V}_d \tag{3}$$

$$\sum_{j\in\mathcal{V}} x_{ij}^{kd} + \sum_{j\in\mathcal{V}} x_{ji}^{kd} = 2y_i^{kd} \quad i\in\mathcal{V}_c, k\in\mathcal{K}, d\in\mathcal{V}_d \tag{4}$$

$$\sum_{i\in\mathcal{V}} x_{ij}^{kd} = \sum_{i\in\mathcal{V}} x_{ji}^{kd} \quad j\in\mathcal{V}, k\in\mathcal{K}, d\in\mathcal{V}_d \tag{5}$$

$$y_d^{kd} \leq \sum_{(i,j)\in\mathcal{A}} x_{ij}^{kd} \quad k\in\mathcal{K}, d\in\mathcal{V}_d \tag{6}$$

$$2y_d^{kd} \leq \sum_{j\in\mathcal{V}_c} x_{jd}^{kd} + \sum_{j\in\mathcal{V}_c} x_{dj}^{kd} \quad k\in\mathcal{K}, d\in\mathcal{V}_d \tag{7}$$

$$\sum_{i\in\mathcal{V}_d}\sum_{j\in\mathcal{V}_c}\sum_{k\in\mathcal{K}} z_{ij}^k = \sum_{j\in\mathcal{V}_c} q_j \tag{8}$$

$$\sum_{i\in\mathcal{V}} z_{ij}^k - \sum_{i\in\mathcal{V}} z_{ji}^k = \sum_{d\in\mathcal{V}_d} q_j y_j^{kd} \quad j\in\mathcal{V}_c, k\in\mathcal{K} \tag{9}$$

$$z_{ij}^k \leq \sum_{d\in\mathcal{V}_d} \left(Q^k - q_i\right) x_{ij}^{kd} \quad i\in\mathcal{V}, j\in\mathcal{V}_c, k\in\mathcal{K} \tag{10}$$

$$x_{ij}^{kd}\in\{0,1\}, y_i^{kd}\in\{0,1\}, z_{ij}^k \geq 0 \quad i,j\in\mathcal{V}, k\in\mathcal{K}, d\in\mathcal{V}_d \tag{11}$$

The objective function (1) minimizes the total cost composed of fixed vehicle costs and variable routing costs. Equations (2) enforce that each customer must be visited exactly once. Constraints (3) impose that if a customer is served by vehicle k housed at depot d, then vehicle k must leave the depot. Constraints (4) and (5) guarantee the flow conservation. Constraints (6) and (7) ensure that a customer i is served by vehicle k only if vehicle k housed at depot d is used. Constraints (8)–(10) impose both the connectivity of the solution and the vehicle capacity constraints. In particular, constraints (8) ensure that each customer demand is satisfied. Summing up these constraints yields constraint (9) which states that the total load leaving all depots must be equal to the total customer demands. Constraints (10) bound the load on each arc (i,j), i.e., after visiting node i the load on arc (i,j) plus the demand of node i cannot exceed the capacity of the vehicle used. Constraints (11) are the non-negativity constraints.

4 Hybridized Adaptive large neighborhood search

The MDFSMOVRP is an NP-Hard problem as its reduction the MDFSMVRP is NP-Hard [4], which justifies the limited capacity of exact methods in solving large instances. Therefore, an heuristic method becomes necessary to solve larger instances. In this section, we propose an hybridized Adapative Large Neighborhood Search (ALNS) metaheuristic embedded with local search procedures. The ALNS was firstly proposed in 2006 by [10] and it is based on the combination of several destroy and repair operators. The idea of the ALNS is to perturbate the solution by destroying a part of it then re-inserting the part destroyed in a better place in a way that improves the overall solution. The choice of the removal and insertion operators is controlled by a roulette wheel mechanism. The principle is the following: for each operator we choose a score and a weight, the choice depends on the past performance of those operators (a very good score, an average score and a low score). At each iteration, the scores are reset. The stop criterion can be the resolution time and/or the number of iterations. We have also developed three local search procedures described in section 4.2 and which aim to improve the solution found by ALNS.

The algorithm can be initialized by a feasible initial solution which can be randomly generated. Usually, the choice of the initial solution does not influence the performance of the ALNS as many iterations will be able to improve the initial solution, [1, 7]. In the proposed method, we randomly assign a depot and a vehicle with a given type to each customer and we create one direct trip per customer to build the initial solution considering that the fleet is unlimited.

4.1 Destroy and repair methods

The objective of destroy operator is to diversify the search while maintaining the promising parts of the incumbent solution, without spending too much computational time. Each destroy operator removes q customers. On the other hand, the insertion operators aim to re-insert the removed customers back in their original routes, or in other routes originating from their original depot, or in any other route starting from a different depot. The proposed metaheuristic is based on five destroy operators and three repair operators, described as follows.

. *Random removal:* This operator randomly selects a number of customers to be removed.
. *Worst customer removal:* The worst customer removal aims to remove the most costly customers. The idea is to select a number of customers that increase the total traveled distance.
. *Worst vehicle removal:* The idea of this heuristic is to remove the customers served by the most costly vehicle.
. *Last customer removal:* This heuristic aims to remove the last customer visited in all routes.
. *Cluster removal:* The cluster removal heuristic aims to remove a set of customers that are similar with respect to a predefined similarity criterion.
. *Random insertion heuristic:* This heuristic randomly inserts a customer in one route. We randomly select a depot and a vehicle, and we insert the customer at a random feasible position.
. *Basic greedy insertion heuristic:* This heuristic inserts a customer in the best possible position of a route.
. *Regret insertion heuristic:* The regret heuristic tries to circumvent this problem by considering more relevant information when selecting the customer to insert.

4.2 Local search

1-0 exchange: This heuristic tries to improve the solution by removing a customer from one route and cost-effectively inserting it in the best position in the same or another route.

2-opt heuristic: The 2-opt heuristic exchanges two pairs of arcs in the same route or in two distinct routes in a way that it reduces the total cost of the solution. The process repeatedly applies 2-opt swaps by systematically testing all possible pairs of arcs until no further improvement is possible.

3-opt heuristic: The 3-opt heuristic is a 3-swap improvement heuristic. It is achieved by replacing three arcs by three new ones without modifying the orientation of the route.

5 Computational experiments

In this section, we provide details on the implementation and we report the results provided by the commercial solver and the ones obtained by the hybrized ALNS. The algorithm was coded with C++ and executed on an "Intel Core i3-6006U with 4 GB of RAM". A large data set of 21 benchmark instances are presented. Instances are firstly solved with IBM CPLEX Concert Technology 12.5.1. We set the time limit of one hour on running CPLEX for each instance. For the proposed metaheuristic, we fix the maximum number of iterations to 3.000.000 for each run for each instance.

5.1 CPLEX results

Table 1 displays the characteristics of the instances considered and presents the results provided by CPLEX. We consider the open version of the MDFSMVRP instances used in [4, 11, 14]. Let N be the number of customers and P the number of depots. The instances contain between 10 and 100 customers, and between two and five depots. Five types of vehicles are considered in all

Table 1. CPLEX results on MDFSMOVRP instances

Instance	N	P	UB	LB	Gap	CPU
1-4-55	55	4	1416.15	1277.41	9.80	3601
2-3-85	85	3	2251.74	1914.63	14.97	3602
3-3-85	85	3	1592.07	1329.01	16.52	3601
4-4-50	50	4	1619.28	1283.72	20.723	3601
5-4-50	50	4	1158.70	859.55	25.82	3601
6-5-75	75	5	1955.42	1383.16	29.27	3602
7-2-100	100	2	2588.18	1968.51	23.94	3602
8-2-100	100	2	1849.68	1291.93	30.15	3601
9-3-100	100	3	2659.23	1941.77	26.98	3602
10-4-100	100	4	2658.21	1935.64	27.18	3603
15-2-80	80	2	2423.61	1629.19	32.78	3601
Average					23.47	3601.55
c-2-10-60	10	2	382.02	382.02	0.00	1
c-2-15-60	15	2	587.88	587.85	0.01	33
c-3-20-80	20	3	592.19	592.13	0.01	729
c-3-25-80	25	3	697.76	697.69	0.01	1689
c-3-30-80	30	3	848.43	809.16	4.63	3608
p-2-10-60	10	2	410.55	410.55	0.00	1
p-2-15-60	15	2	616.27	616.21	0.01	119
p-3-20-100	20	3	515.76	515.73	0.01	148
p-3-25-100	25	3	624.46	624.40	0.01	1299
p-3-30-100	30	3	769.20	766.58	0.34	3601
Average					0,50	1122,8

instances. For each instance, we provide the bounds found by CPLEX in columns headed UB and LB. We also report the average gap and the running time in seconds in the last two columns.

Results in Table 1 report the average gap of 23,47% for large instances and the average gap of 0,50% for small instances. CPLEX proves the optimality for 2 instances and found an average gap of 0.01% for 6 instances. One can conclude that on small instances, CPLEX performs very well, however on large instances, CPLEX may be less efficient with an average gap that can reach 32% for large-sized instances with 2 depots and 80 customers.

5.2 Hybridized ALNS results

Table 2 reports the results provided by the proposed metaheuristic on open MDFSMOVRP instances. For each instance, we report the best cost found over two runs, the number of vehicles used NV, the best running time in seconds and the average gap compared to the lower bound provided by CPLEX and calculated as follows: $Gap = \frac{100*(Cost_{ALNS}-UB_{CPLEX})}{Cost_{ALNS}}$.

Obviously, the running time of the proposed matheuristic to solve all instances is considerably lower than the running time of CPLEX with an average equals to 113,62 seconds. The hybridized ALNS was able to find two optimal solutions as CPLEX and improves 8 best known solutions provided by CPLEX and which are highlighted in bold in Table 2. The average gap is low on small instances but it is considerably high on larger instances. Indeed, for some large instances with 3 or more depots (e.g., 1-4-55, 2-3-85, p-3-25-100, p-3-30-100) the hybridized ALNS was not able to improve the results provided by CPLEX. A solution to overcome this drawback is to increase the maximum number of iterations as the proposed method is very fast.

6 Conclusions

In this paper we have studied the MDFSMOVRP, an extension of several classical problems such as the MDOVRP, the MDFSMVRP and the FSMOVRP. We have proposed a mathematical model and an hybridized ALNS to solve the problem. As a future work, we may consider to test the proposed metaheuristic on those reduced versions and to conduct some parameter tuning to enhance its performance.

Table 2. Computational results provided the metaheuristic on MDFSMOVRP instances

Instance	N	P	$Cost_{ALNS}$	NV	CPU	Gap
1-4-55	55	4	1568.51	13	126	18.56
2-3-85	85	3	2292.65	17	193	16.49
3-3-85	85	3	1690.87	14	196	21.40
4-4-50	50	4	**1497.75**	10	123	14.29
5-4-50	50	4	**962.52**	6	118	10.70
6-5-75	75	5	**1675.45**	13	224	17.45
7-2-100	100	2	**2340.65**	17	207	15.90
8-2-100	100	2	**1499.64**	10	201	13.85
9-3-100	100	3	**2309.54**	16	254	15.92
10-4-100	100	4	**2315.78**	17	290	16.42
15-2-80	80	2	**1916.50**	14	146	14.99
c-2-10-60	10	2	382.02	2	14	0.00
c-2-15-60	15	2	593.04	3	20	0.88
c-3-20-80	20	3	624.05	4	34	5.11
c-3-25-80	25	3	745.05	5	42	6.36
c-3-30-80	30	3	867.64	5	51	6.74
p-2-10-60	10	2	410.55	3	13	0.00
p-2-15-60	15	2	622.19	4	19	0.96
p-3-20-100	20	3	551.96	4	30	6.56
p-3-25-100	25	3	716.57	5	38	12.86
p-3-30-100	30	3	871.82	6	47	12.07
Average				8.95	113,62	10.83

References

[1] D. Aksen, O. Kaya, F. S. Salman, and Ö. Tüncel. An adaptive large neighborhood search algorithm for a selective and periodic inventory routing problem. *European Journal of Operational Research*, 239(2):413–426, 2014.

[2] L. Guezouli and S. Abdelhamid. A multi-objective optimization of multi-depot fleet size and mix vehicle routing problem with time window. In *6th International Conference on Systems and Control*, pages 328–333. IEEE, 2017.

[3] S. Irnich. A multi-depot pickup and delivery problem with a single hub and heterogeneous vehicles. *European Journal of Operational Research*, 122(2):310–328, 2000.

[4] R. Lahyani, L. Coelho, and J. Renaud. Alternative formulations and improved bounds for the multi-depot fleet size and mix vehicle routing problem. *OR Spectrum*, 40(1):125–157, 2018.

[5] E. Lalla-Ruiz, C. Expósito-Izquierdo, S. Taheripour, and S. Voß. An improved formulation for the multi-depot open vehicle routing problem. *OR spectrum*, 38(1):175–187, 2016.

[6] R. Liu, Z. Jiang, and N. Geng. A hybrid genetic algorithm for the multi-depot open vehicle routing problem. *OR Spectrum*, 36(2):401–421, 2014.

[7] M. A. Pereira, L. A. N. Coelho, L.and Lorena, and L. C. De Souza. A hybrid method for the probabilistic maximal covering location–allocation problem. *Computers & Operations Research*, 57:51–59, 2015.

[8] K. Pichka, B. Ashjari, A. Ziaeifar, and P. Nickbeen. Open vehicle routing problem optimization under realistic assumptions. *International Journal of Research in Industrail Engineering*, 3 (2), 2014.

[9] C. Y. Ren. Research on improved genetic algorithm for heterogeneous open vehicle routing problem. In *Applied Mechanics and Materials*, volume 55, pages 859–862. Trans Tech Publ, 2011.

[10] S. Ropke and D. Pisinger. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40(4):455–472, 2006.

[11] S. Salhi, A. Imran, and N. A. Wassan. The multi-depot vehicle routing problem with heterogeneous vehicle fleet: Formulation and a variable neighborhood search implementation. *Computers & Operations Research*, 52:315–325, 2014.

[12] M. Soto, M. Sevaux, A. Rossi, and A. Reinholz. Multiple neighborhood search, tabu search and ejection chains for the multi-depot open vehicle routing problem. *Computers & Industrial Engineering*, 107:211–222, 2017.

[13] C. D. Tarantilis and C. T. Kiranoudis. Distribution of fresh meat. *Journal of Food Engineering*, 51(1):85–91, 2002.

[14] T. Vidal, T. G. Crainic, M. Gendreau, and C. Prins. Implicit depot assignments and rotations in vehicle routing heuristics. *European Journal of Operational Research*, 237(1):15–28, 2014.

[15] B. Yao, P. Hu, M. Zhang, and X. Tian. Improved ant colony optimization for seafood product delivery routing problem. *PROMET-Traffic & Transportation*, 26(1):1–10, 2014.

[16] M. Yousefikhoshbakht, F. Didehvar, and F. Rahmati. Solving the heterogeneous fixed fleet open vehicle routing problem by a combined metaheuristic algorithm. *International Journal of Production Research*, 52(9):2565–2575, 2014.

[17] M. Yousefikhoshbakht, F. Didehvar, and F. Rahmati. A mixed integer programming formulation for the heterogeneous fixed fleet open vehicle routing problem. *Journal of Optimization in Industrial Engineering*, 8(18):37–46, 2015.

[18] M. Yousefikhoshbakht, A. Dolatnejad, F. Didehvar, and F. Rahmati. A modified column generation to solve the heterogeneous fixed fleet open vehicle routing problem. *Journal of Engineering*, 3, 2016.

Evolutionary operators in memetic algorithm for matrix tri-factorization problem

Rok Hribar[1,2], Gašper Petelin[3], Jurij Šilc[1], Gregor Papa[1,2], and Vida Vukašinović[1]

[1] Jožef Stefan Institute, Jamova cesta 39, 1000 Ljubljana, Slovenia
rok.hribar@ijs.si
jurij.silc@ijs.si
gregor.papa@ijs.si
vida.vukasinovic@ijs.si
[2] Jožef Stefan International Postgraduate School, Jamova 39, 1000 Ljubljana, Slovenia
[3] Faculty of Computer and Information Science, University of Ljubljana, Večna pot 113, 1000 Ljubljana, Slovenia
gapi.petelin@gmail.com

Abstract

In memetic algorithm, a population based global search technique is used to broadly locate good areas of the search space, while repeated usage of a local search heuristic is employed to locate optimum. Intuitively, evolutionary operators that generate individuals with genetic material inherited from the parents and improved performance ability should be the right option for improved performance of the algorithm in terms of time and solution quality. Evolutionary operators with such properties were devised and used in memetic algorithm for solving multi-objective matrix tri-factorization problem. It was shown, by comparing deterministic naive approach with two variants of memetic algorithm with different level of inheritance, that evolutionary operators do not improve performance in this case. Further analysis showed that even though proposed evolutionary operators inherit high fitness from its parents, local search does not perform well on such offspring which results in poor performance.

Keywords: Memetic algorithms, Non-negative matrix factorization, Multi-objective optimization, Gradient descent

1 Introduction

The level of data generated within different area of our life has drastically increased with the expand of information technology. As a consequence, an interests to extract meaningful information out of collected data significantly grew and the knowledge discovery has become widely studied research area. Within this research, we try to understand data by forming groups of instances, i.e. clusters, where instances in the same cluster are in some sense more similar to each other than the instances in other clusters. The problem studied in this paper is non-negative matrix factorization (NMF) problem which generalizes kernel k-means clustering, bipartite graph k-means clustering and spectral clustering problem [1]. Original NMF factorizes input non-negative matrix R into two non-negative matrices so that $R \approx GQ^T$, where $R \in \mathbb{R}_+^{n\times m}, G \in \mathbb{R}_+^{n\times k}$, and $Q \in \mathbb{R}_+^{m\times k}$. NMF's main objective is clustering of columns of R. While NMF can capture two types of relations, non-negative matrix tri-factorization (NMTF) $R \approx GSQ^T$ can capture more types of information [2]. Both approaches are used for revealing hidden patterns in large real-world datasets and give a good framework for simultaneously clustering the rows and columns of R. NMTF problem can be encountered in image processing, text mining, hyperspectral unmixing and bioinformatics [3, 7, 8]. Additionally, Buono and Pio proved that NMTF has several advantages compared to the original NMF approach [4].

This paper is concerned with the behavior of evolutionary operators used in solving NMTF problem with presence of additional local search strategies. In Evolutionary algorithm (EA), mechanisms inspired by biological evolution such as selection, crossover and mutation influence the evolution of population and implicitly lead the performance of evolutionary search. Mitchell and Holland analyzed promising features of genetic algorithm for its speedup and suggested that crossover in idealized genetic algorithm should create instances with higher fitness [5]. Doerr et al. provided first theoretical proof for usefulness of crossover for non-artificial problem [6]. In this work, we develop a mutation and crossover operator which, applied on matrices, are able to provide solutions with lower objective values and we compare memetic algorithms (with and without suggested crossover) and deterministic naive approach on non-negative matrix tri-factorization problem (NMTF).

2 Multi-objective non-negative matrix tri-factorization problem

The aim of NMTF is to extract insights of intra-relations of some data set. If the intra-relations are expressed by non-negative symmetric matrix R, then by NMTF of the form $R = GSG^T$, where G and S are non-negative matrices of dimensions much smaller than dimension R, some insights how data is clustered and what are relations among those clusters can be provided. Further, in a co-clustering version of NMTF problem a set of matrices R_i needs to be factored using the same G as it is shown in Eq. (1).

$$\forall i : \quad \boxed{R_i} = \boxed{G} \cdot \boxed{S_i} \cdot \boxed{G^T} \tag{1}$$

Here, the columns of G can be interpreted as clusters, while components of S_i can be interpreted as interactions among these clusters.

This problem can be stated as an optimization problem by minimizing relative square error

$$\mathrm{RSE} = \frac{\sum_i \|R_i - GS_iG^T\|_F^2}{\sum_i \|R_i\|_F^2}, \tag{2}$$

where $\|\cdot\|_F$ is the Frobenius norm. Note, that optimal solution has $RSE = 0$, while trivial solution ($G = 0$ and $S_i = 0$) has $RSE = 1$. Given that G matrix is common to all R_i tri-factorizations, all dimensions of S_i are the same. This dimensions' number can be interpreted as the number of clusters and it is not known in advance. In order to assure high ability of data relationships' interpretation, the second objective is to minimize

$$k = \dim S_i. \tag{3}$$

In this respect, RSE minimization ensures the accuracy of tri-factorization and k minimization ensures that the size of representation is as small as possible. Note that the objectives in Eq. (2) and (3) are contradictory because the capacity of GSG^T model grows with k.

3 Naive approach

If k is fixed, RSE can be minimized via gradient descent since RSE from Eq. (2) is a differentiable function of G and S_i. Libraries for automatic differentiation such as `tensorflow`, `theano` or `CNTK` can be used to calculate the gradient of RSE with respect to G and S_i and update them in direction of the gradient. It must be stressed that the second objective k is not differentiable, hence gradient descent can only be used to minimize one objective.

In this work, special version of gradient descent algorithm called Adam [9] is used. This algorithm is well suited for large problems and has two main benefits. One is is the adaptive learning rate control which changes step size during descent in response to changes in gradient magnitude. The second benefit is the use of momentum which prevents oscillations in narrow valleys of the search space and gives the descent an ability to skip shallow local minima. Both traits reduce the number of steps needed to find a local minimum.

The non-negativity constraint encountered in problem definition can be easily fulfilled if absolute value is applied to G and S_i before every RSE calculation. In this way a step of gradient descent going through the bound of the feasible region is effectively bounced back to the feasible region.

A stopping criterion for gradient descent was devised where relative differences in objective function among successive steps are used as an indicator of convergence. Median of the several past differences was found to be a reliable estimate of the pace of convergence.[4] When this pace falls far below previously encountered ones the gradient descent is stopped. Additionally, descent is also stopped if maximum number of steps is exceeded.

Even though k is not differentiable, it is possible to solve the two-objective optimization problem using only gradient descent. In naive approach (NA) Adam is performed for many different k until satisfactory approximation of Pareto front is acquired. However, there are no guidelines how large the desired k might be and the search can be concentrated to a region where k is too small. In such cases, valuable computational time is being wasted.

4 Memetic algorithm

The term memetic algorithm is used to describe a synergetic combination of an evolutionary approach and local improvement procedure. In this work memetic algorithm for above described NMTF problem is developed with the main motivation to benefit from the combination of good hereditary features of EA and efficient Adam, which is used as local improvement procedure. Stopping criteria for Adam are the same as in naive approach.

4.1 Evolutionary algorithm

Standard EA operators are adapted in such a way that offspring inherit good traits from its predecessors, i.e. no matter what is the dimension of S, a comparably low RSE value is ensured. The aim of EA is to provide good starting individuals with various dimension k for further treatment with Adam, which is usually able to further decrease RSE. In this manner evolutionary algorithm is used only to find good initial points for gradient descent which should reduce the computational load to a high degree.

The difference between suggested adapted EA and the classical one is that in the suggested algorithm the evolutionary process starts with a very small initial population which is growing linearly over time. As a selection of individuals for breeding, a tournament selection is used. We privilege individuals, where Adam was successful, hence the criteria to win the tournament is the lowest RSE. Evolutionary search is also slowly switched from exploration to exploitation; in the beginning parents are selected at random, while in the later generations individuals with lower RSE are preferably chosen to become parents. This is accomplished by setting the proportion between the tournament size and the number of individuals in the population constant over all generations.

Crossover operator used in this work combines two parents and produces one offspring. Offspring's G matrix is a concatenation of parents' G matrices along rows, while offspring's S_i matrices are a direct sum of parents' S_i matrices, see Fig. 1 for an illustration. Note, that by crossover operator individuals with enlarged dimension k are obtained but it holds that

$$\mathrm{RSE}(\text{offspring}) \leq 1/2\,(\mathrm{RSE}(\text{parent1}) + \mathrm{RSE}(\text{parent2})). \tag{4}$$

In order to prove statement (4) it is sufficient to show that this inequality holds for a single summand in Eq. (2). Let M_1, M_2 be GSG^T-products of the parents, while the offspring's GSG^T-product is $1/2(M_1 + M_2)$ by definition. Using this fact, it follows

$$\begin{aligned} \|R - 1/2(M_1 + M_2)\|^2 &\leq 1/4\,(\|R - M_1\| + \|R - M_2\|)^2 && (5)\\ &\leq 1/2\left(\|R - M_1\|^2 + \|R - M_2\|^2\right), && (6) \end{aligned}$$

where in (5) triangle inequality and in (6) the fact that $2xy \leq x^2 + y^2$ was used[5]. Clearly, an offspring inherits comparable low RSE value from its parents.

Mutation operator used in this work either deletes or adds columns to matrix G and corresponding rows and columns to matrices S_i, see Fig. 2 for an illustration. Columns and rows added by mutation are populated with small random values. In case of deletion, columns and rows that contribute the least to the $\|GS_iG^T\|_F$ are chosen. If columns of G are normalized, then by inspecting the smallest values of S_i components it is easy to determine which columns contribute

[4] Mean was found to be too susceptible to outliers which are also encountered during gradient descent.

[5] $2xy \leq x^2 + y^2$ is equivalent to $0 \leq (x - y)^2$

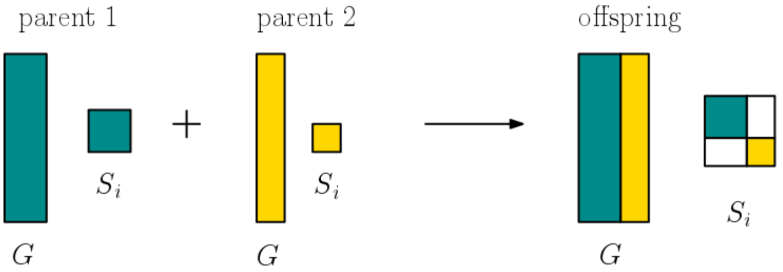


Fig. 1: Crossover of two parents.

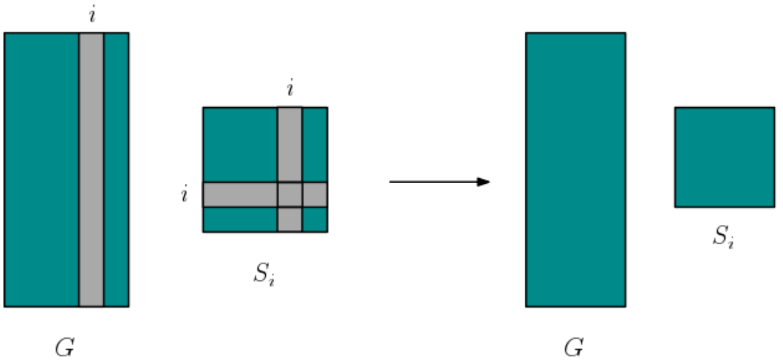


Fig. 2: Mutation where deletion of one column of G was applied.

the least. In this regard, those rows and columns of S_i and corresponding ones in G are deleted that are least significant. By applying a mutation on the selected individual, new offspring with different k and minimally altered RSE is obtained.

New individuals, constructed by crossover and mutation operators, are improved using Adam algorithm at the end of every generation.

Offspring created by mutation or crossover both inherit good RSE from their predecessors. In this regard, the main purpose of proposed evolutionary operators is their ability to construct good initial points for gradient descent from already descended individuals from the population. By using evolutionary operators, information about good clusters and their interactions is able to flow across individuals that have matrices of different dimensions. More importantly, good clusters found in lower dimensions can be passed to higher dimensional individuals for which computational load of using Adam is more pronounced. By passing this information to large individuals, the number of gradient descent steps should be reduced to a large degree.

Two versions of memetic algorithm were used in this work. The M1, performs only mutations and the second, M2, performs both crossovers and mutations.

5 Experiments

A test problem was constructed with 5 matrices R_i of dimension 800 that have a known minimum at RSE $= 0$ and $k = 50$. Approximately 1/3 of R_i components were non-zero and their magnitude was around one. Three algorithms were used to solve this problem, i.e. M1, M2, and NA. Each algorithm was run 12 times due to limited computational resources.

Basic component of all algorithms in this work is Adam. The parameters of Adam algorithm were manually tuned beforehand, starting with values proposed in the literature [9]. The optimal parameters found were $\alpha = 0.001$, $\beta_1 = 0.9$ and $\beta_2 = 0.99$ where notation from [9] is assumed. In order to ensure reasonable execution time, maximum number of steps for Adam was chosen to be 5000. Convergence criterion was fulfilled when median of last 150 relative differences dropped below one third of the worst seen median. This convergence criterion was devised by observing how gradient descent progresses for this type of problems. Initial experiments showed that when this criterion is fulfilled, there is a small probability that continuing gradient descent will bring further improvements.

Initial matrices G and S_i for all algorithms were chosen randomly where matrix components were drawn from uniform distribution on interval $[0, 0.01]$. In this way initial matrices are close to the trivial solution ($G = 0$ and $S_i = 0$) with RSE ≈ 1. Initial experiments have shown that taking larger initial components of the matrices results to a larger number of steps before gradient descent converges.

Algorithm NA started with $k = 1$ and incremented k by one in each generation. For each k Adam is executed starting from initial random matrices. NA was stopped when it encountered an individual with RSE < 0.01.

Both M1 and M2 started with a population of 4 individuals whose k was chosen from a uniform distribution on set $\{1, 2, \ldots, 7\}$. M1 preforms only mutations, while M2 preforms both crossovers and mutations. For each crossover M2 performs two mutations. The number of columns deleted or added during mutation was drawn from geometric distribution with expected value equal to 3.0. At the end of each generation gradient descent was performed on all new individuals. Crossover of a parent with itself was prevented due to the inability of gradient descent to improve such an offspring. The stopping criterion is the same as for NA which is fulfilled when RSE < 0.01 for some individual in the population.

6 Results

A comparison was done among M1, M2 and NA algorithms with regard to the hypervolume and the number of evaluations. Fig. 3 and 4 depict the distributions of these two indicators. A depiction of Pareto front approximations obtained over all runs by each algorithm is shown if Fig. 5.

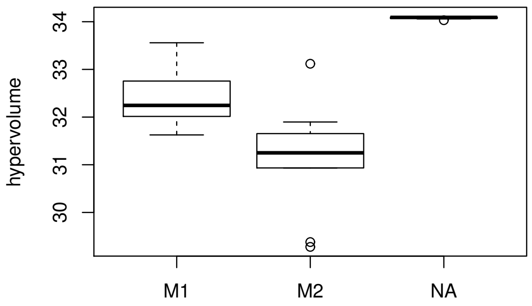


Fig. 3: Box plot of hypervolumes for 12 runs of algorithms M1, M2 and NA.

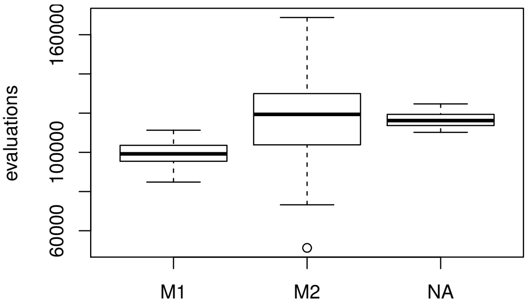


Fig. 4: Box plot of number of evaluations for 12 runs of algorithms M1, M2 and NA.

The number of runs is rather small for statistical comparison, however it can give indications for further work. The data for both comparisons was analyzed using the traditional approach [10] and the Deep Statistical Comparison (DSC) approach [11]. Using the traditional approach, the normality condition (checked with the one-dimensional Kolmogorov-Smirnov test for normality) with regard to the number of evaluations was satisfied, while with regard to the hypervolume was not satisfied. For both comparisons, the homoscedasticity of the variance was checked with Levene's test and for both comparisons the condition was not satisfied. For this reason, the Kruskal-Wallis test was selected as an appropriate omnibus statistical test.

With regard to the hypervolume, there is a statistical significance between the three algorithms, and this significance comes from the difference between the pairs M1, NA and M2, NA. For the same comparison the DSC approach showed that there is a statistical significance between the three algorithms M1, M2 and NA, and they are ranked as 2, 3, and 1, respectively, which can also be seen in Fig. 3.

Regarding the number of evaluations, the Kruskal-Wallis test showed there is a statistical significance between the three algorithms, however the post hoc test according to Dunn showed that the significance comes with regard to the pairs M1, M2 and M1, NA, while there is no difference

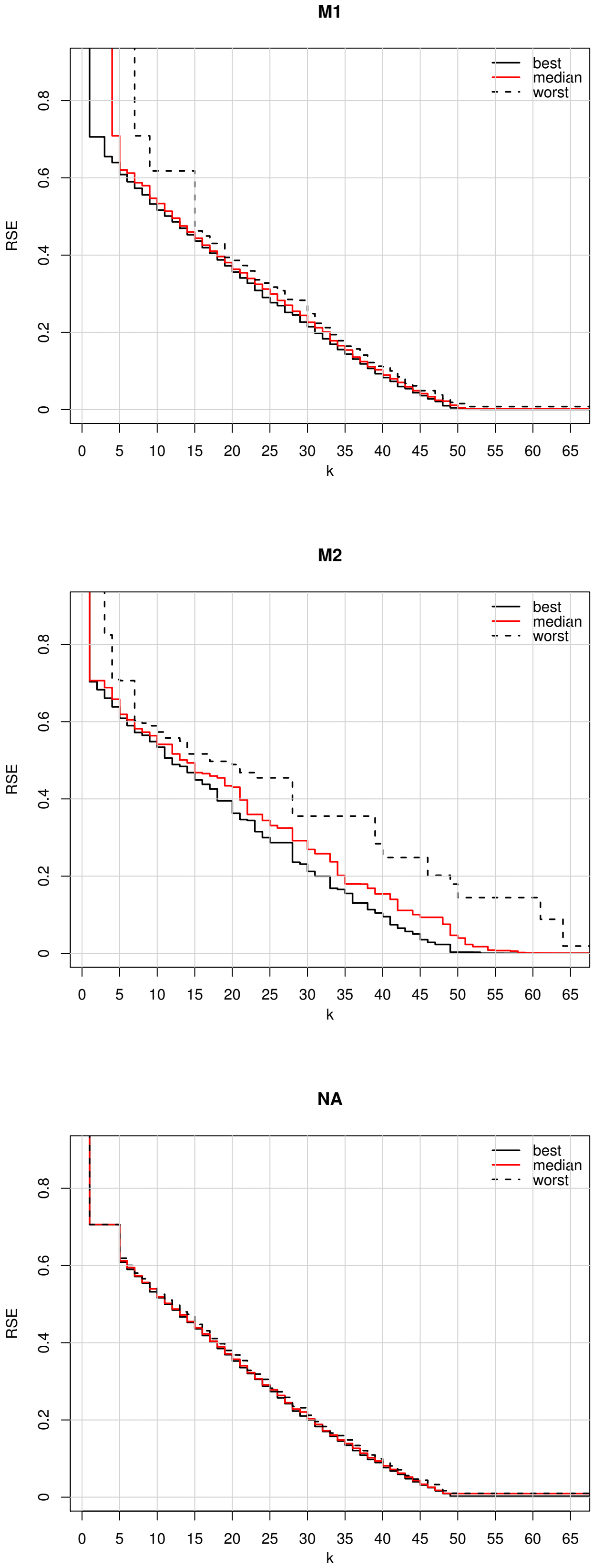


Fig. 5: Summary attainment surfaces of Pareto fronts over 12 runs for algorithms M1, M2 and NA.

between M2, NA. However the traditional approach with Kruskal-Wallis test is made with regard to the medians and not taking into account different standard deviation of the distribution. For this reason, a recently proposed DSC approach was used, where the comparison is made taking into account the whole distribution of the data. The result is that there is a statistical significant difference among M1,M2 and NA, and they are ranked as 1, 2, and 3, respectively, from which it follows that the NA needs most evaluations on average.

The fact that NA finds better quality solutions compared to M1 and M2 is very surprising. Even though evolutionary operators generate initial points with lower values of RSE, it seems that those initial points do not lead gradient descent to good regions. Evidently, starting with low RSE does not guarantee good convergence. This indicates that low RSE should not be the sole trait to be inherited in order to ensure efficient evolutionary operators. To further explore this counterintuitive behavior, the data gathered during optimizations was analyzed. All instances of individuals with $k = 10$ that was generated during M1, M2 or NA was gathered. Such individuals can be produced by crossover or mutation followed by a gradient descent or it can be produced solely by gradient descent starting from random initial point.

Fig. 6 shows the progression of gradient descent for individuals generated by crossover and for randomly generated individuals. Because crossover combines previously optimized parents, the offspring has low initial RSE compared to the random point whose RSE ≈ 1 at the start of gradient descent. When gradient descent is used on individuals created by crossover, RSE steeply falls but shortly after the convergence becomes very slow. It seems that gradient descent enters a region of slow convergence which could indicate a plateau in the optimization landscape. On the other hand, when gradient descent starts from random initial point, the convergence is quite even and no quick changes in steepness are present. It seems that crossover introduces such initial points for gradient descent that are drawn to a plateau with very small gradient.

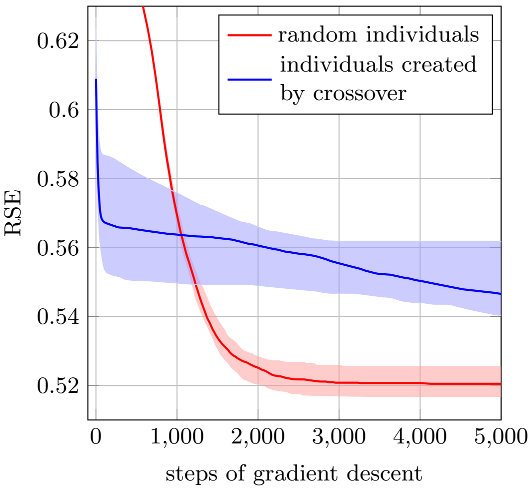


Fig. 6: Progress of gradient descent when initial points are individuals created by crossover compared to the random individuals. All solutions here have $k = 10$ and crossovers were performed using already descended individuals with $k = 3, 4, 5, 6, 7$ $(3 + 7,\ 4 + 6$ and $5 + 5)$. Full lines are the medians and the shaded areas represent the range of central 66% of runs.

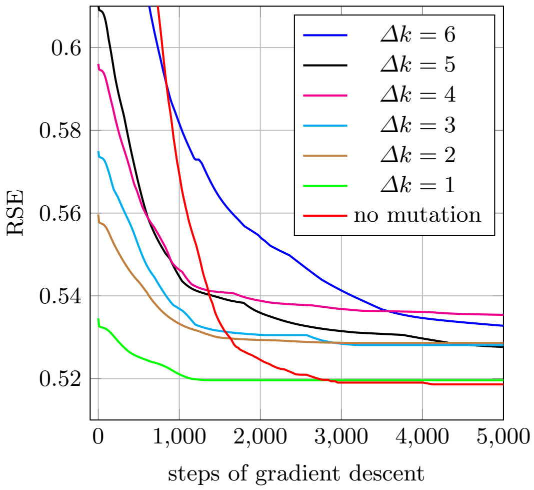


Fig. 7: Median progress of gradient descent when initial points are random individuals compared to mutated individuals. All solutions here have $k = 10$ and mutations were performed using already descended individuals with $k = 4, 5, \ldots, 9$. Quantity Δk tells how many columns were added via mutation and represents the extent to which an individual was mutated.

Fig. 7 shows the progressions of gradient descent for individuals generated by mutation. Only mutations where columns were added were considered. The extent to which an individual is mutated is measured by the number of columns that was added to an individual Δk. Gradient descent performed on mutated individuals converges to higher values of RSE compared to the one performed on random individuals. The only mutation that leads gradient descent to RSE values close to the ones in nonmutated case is the addition of one column ($\Delta k = 1$). The number of steps needed to reach convergence in this case is also approximately three times smaller compared to nonmutated case. This is one explanation why M1 requires less evaluations compared to M2 and NA. The distributions of RSE values after gradient descent for mutated individuals is shown in Fig. 8. Like with crossover, the mutation seems to lead gradient descent to unfavorable regions of the search space.

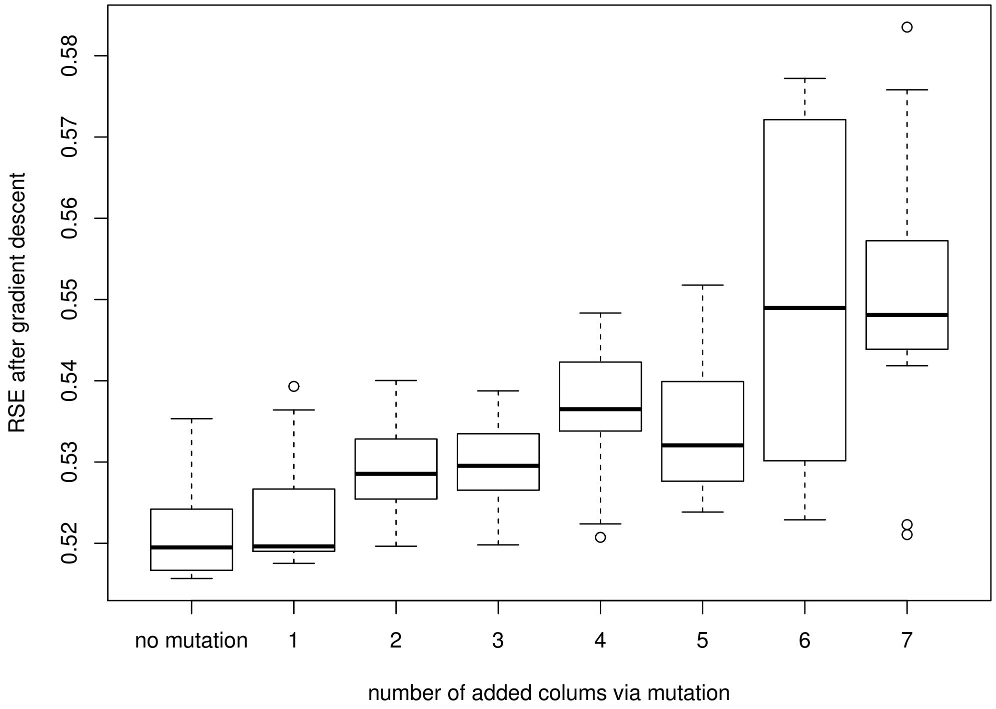


Fig. 8: Distributions of RSE after performing gradient descent on mutated individuals compared to non-mutated ones. Mutation considered here is the addition of specific number of columns (x axis) to matrices G and S_i to already descended individuals. All solutions here have $k = 10$, therefore mutations were performed on already descended individuals with $k = 3, 4, \ldots, 9$.

7 Conclusion and future work

The comparison of the three algorithms showed that the naive approach consistently surpasses memetic algorithms in the quality of solutions. Even though, memetic algorithms can generate individuals which inherit low RSE, these offspring do not lead Adam to solutions of the same quality compared to random starting points. This was further proved by analyzing gradient descent progress of individuals that were generated via crossover, mutation or generated randomly. This can indicate an interesting property of the fitness landscape that needs to be further explored. In particular, Adam in memetic algorithms starts on matrices with few dominant non-zero entries provided by evolutionary operators, while in naive approach Adam starts its search from matrices with entries which are uniformly distributed between $[0, 0.01]$. This might be the reason for low local search performance and might indicate that the choice of evolutionary operators should also be dependent of the local search used and not only on good hereditary features. Also, it seems that inheritance of traits from good individuals of lower dimension somehow guides the gradient descent to regions where gradient has very small magnitude.

Non the less, memetic algorithms proved to be significantly faster than the naive approach.

For the future work we will continue our studies of efficient evolutionary operators for the research problem. Since further testing of evolutionary operators behavior for this problem requires great computational resources, acceleration of gradient descent step will be implemented on GPGPU.

Evolutionary algorithm presented in this work will be further developed so that matrices G have orthogonal columns. This constraint restricts the problem and enforces classical interpretation of clustering. This algorithm will also be adapted so that asymmetric matrices R_i can be used which in consequence introduces several different G matrices of different dimensions, one for each data type. In this regard, the algorithm will be able to perform both classical and soft clustering for possibly heterogeneous data.

Acknowledgements

We would like to thank Tome Eftimov for guidance in statistical comparison of obtained results and Tea Tušar and Peter Korošec for useful conversations. This research was partially supported by Slovenian Research Agency program No. P2-0098, project No. J1-8155 and project No. PR-07606. It is also partially founded by the European Union's Horizon 2020 research and innovation program under Grant Agreement No. 692286.

References

1. Chris Ding, Xiaofeng He, and Horst D Simon. On the equivalence of nonnegative matrix factorization and spectral clustering. In *Proceedings of the 2005 SIAM International Conference on Data Mining*, pages 606–610. SIAM, 2005.
2. Chris Ding. Orthogonal nonnegative matrix tri-factorizations for clustering. In *In SIGKDD*, pages 126–135. Press, 2006.
3. Yulong Pei, Nilanjan Chakraborty, and Katia Sycara. Nonnegative matrix tri-factorization with graph regularization for community detection in social networks. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, pages 2083–2089. AAAI Press, 2015.
4. N. Del Buono and G. Pio. Non-negative matrix tri-factorization for co-clustering: An analysis of the block matrix. *Information Sciences*, 301:13 – 26, 2015.
5. Melanie Mitchell and John H. Holland. When will a genetic algorithm outperform hill climbing? In Stephanie Forrest, editor, *Proceedings of the 5th International Conference on Genetic Algorithms, Urbana-Champaign, IL, USA, June 1993*, page 647. Morgan Kaufmann, 1993.
6. Benjamin Doerr, Edda Happ, and Christian Klein. Crossover can provably be useful in evolutionary computation. *Theoretical Computer Science*, 425:17 – 33, 2012. Theoretical Foundations of Evolutionary Computation.
7. Nicolas Gillis. The why and how of nonnegative matrix factorization. *Regularization, Optimization, Kernels, and Support Vector Machines*, 12(257), 2014.
8. Vladimir Gligorijević, Noël Malod-Dognin, and Nataša Pržulj. Integrative methods for analyzing big data in precision medicine. *Proteomics*, 16(5):741–758, 2016.
9. Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
10. Salvador García, Daniel Molina, Manuel Lozano, and Francisco Herrera. A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the cec'2005 special session on real parameter optimization. *Journal of Heuristics*, 15(6):617, 2009.
11. Tome Eftimov, Peter Korošec, and Barbara Koroušić Seljak. A novel approach to statistical comparison of meta-heuristic stochastic optimization algorithms using deep statistics. *Information Sciences*, 417:186–215, 2017.

IP Assignment Optimization for an Efficient NoC-based System using Multi-objective Differential Evolution

Maamar Bougherara[1,2], Rebiha Kemcha[2,3] , Nadia Nedjah[4]
Djamel Bennouar[1] and Luiza de macedo Mourelle[5]

[1] LIMPAF Laboratory, Bouira University, Algeria
bougherara.maamar@gmail.com
[2] Département d'Informatique,
Ecole Normale Supérieure Kouba, Algiers, Algeria
[3] LIMOSE Laboratory, Boumerdes University, Algeria
[4] Department of Electronics Engineering and Telecommunications,
[5] Department of Systems Engineering and Computation,
State University of Rio de Janeiro, Brazil

1 Introduction

Systems-on-Chip (SoC) integrate a complete system into a single chip. The number of integrated components continues to increase to meet the requierments of nowadays applications. Thereby, the design of SoCs focuses in communications more than computations [1]. Network-on-Chip (NoC) emerged as a viable solution to on-chip communication bottlenecks. It is similar to a general network, but with limited resources, power and area.

Each component of the NoC is an Intellectual Proprety (IP) block that can be a processor, memory and DSP [1]. A NoC is designed to run a specific application, which are, generally, limited by the number of the tasks that are implemented in IP blocks. An IP block can implement many tasks and execute them as a general-purpose processor does. In contrast, an IP block is conditioned to execute some types of tasks. Therefore, it is necessary to choose the adequate IP block before designing an efficient NoC-based system for any application. The choice becomes harder when the number of tasks increases. It is also necessary to map these blocks onto the NoC available infrastructure, which consists of a set of cores communicating through switches.

Different optimization criteria can be pursued depending on the detailed information of the application and IP blocks. The application is viewed as a graph of tasks called Task Graph (TG). The features of an IP block can be determined from its companion library [2]. The objectives involved in task assignment and IP block mapping are multiple and have to be optimized simultaneously. Some of these objectives are conflicting because of their nature. So, IP assignment and IP mapping are classified as NP-hard problems[3]. Thus, it is mandatory to use Multi-Objective Evolutionary Algorithms (MOEAs) with specific objective functions.

In this paper, we propose a multi-objective evolutionary strategy to help NoC designers to select the most suited IP blocks chosen during the assignment step. For this aim, we use the structure representation of TG in [4] and IP repository data from the Embedded Systems Synthesis benchmarks Suite (E3S) [5] as an IP library for the proposed tool. We use the Differential Evolution for Multi-objective Optimization (DEMO) algorithm [19], which was modified to suit the specificities of the assignment problem and also to guarantee the NoC designers constraints .

The rest of the paper is organized as follows. First, in Section 2, we present the related work. Subsequently, in Section 3, we present the model used for the application structure and IP repository. Section 4 provides an overview of the assignment problem. Then, in Section 5, we present the Differential Evolution for Multi-objective Optimization algorithm used in this work. After that, in Section 6, we define the objective functions to be optimized by DEMO. There follows, in Section 7, some experimental results and their comparison with some existing works. Finally, in Section 8, we draw some conclusions and present new directions for future work.

2 Related Work

The problems of allocating IP blocks to application tasks and mapping those blocks into a NoC have been addressed in some previous works. Some of these works treat the assignment and mapping as one single

NP-hard problem while others consider them as two distinct NP-hard problems that can be solved separately. Also, some of these works did not take into account the multi-objective nature of the problems and, thus, adopted one single objective. In the following, we describe some works wherein the assignment and mapping problems are handled via multi-objective optimization.

Zhou et al. [6] suggested a multi-objective exploration approach, treating the mapping problem as two conflicting objective optimization problems, that attempts to minimize the average number of hops and achieve a thermal balance. The number of hops is incremented every time data cross a switch before reaching its target. The NSGA multi-objective evolutionary algorithm was used in this case. The authors formulated a thermal model to avoid hot spots, which are areas with high computing activity.

Jena and Sharma [7] addressed the problem of topological mapping of IPs/cores into a mesh-based NoC in two systematic steps using NSGA-II. The main concern was to obtain a solution that minimizes the energy consumption due to both computational and communicational activities and also minimizes the link bandwidth requirements under some prescribed performance constraints.

Da Silva et al. [4] devised an efficient IP assignment multi-objective optimization for NoC platforms. It did so using both NSGA-II and microGA. The work's main concern was the optimization of hardware area, power consumption and execution time.

Radu [8] used three methods to assign tasks into IP blocks: (i) random assignment, wherein many tasks can be assigned to the same IP; (ii) direct assignment, wherein a task is assigned to the first IP that can run it; (iii) minimum assignment, wherein the selection is driven by the minimum run time, and thus it always assigns a task to the IP that runs the fastest. After that, Simulated Annealing is used to perform mapping aiming at minimizing the energy consumption.

In [9], SPEA-II and NSGA-II were used with some different crossover and mutation operators for application mapping. The work aimed at optimizing two objectives, which are energy consumption and thermal balance.

3 Application and IP Repository Models

In order to formulate the IP assignment problem, it is necessary to define the application internal model that will be used. An application can be represented by a set of tasks that can be executed sequentially or in parallel. It is represented by a directed acyclic graph, called Task Graph (TG) [4].

Definition 1 [4]: A task graph $G = (T, D)$ is a directed graph, wherein a node represents a task $t_i \in T$ and a directed arc $d_{ij} \in D$ between tasks t_i and t_j represents the data dependency between these tasks. The arc label $v(d_{ij})$ characterizes the volume of bits exchanged in communication between tasks t_i and t_j.
IP assignment determines the association between each task of the application and the IP block that would execute that task. The result of this step is another graph of IP representing the IPs used to implement the application. This graph is called Application Characterization Graph (ACG).

Definition 2 [4]: An application characterization graph $G = G(C, A)$ is a directed acyclic graph, wherein each vertex $c_i \in C$ represents a selected IP block and each directed arc $a_{ij} \in A$ characterizes the communication process between c_i and c_j.

In order to help NoC designers and standardize the proposed solution, we structured the used application repository, which is the E3S benchmark suite [10] using XML. XML schema provide a neat and well-accepted model for the task graph and IP repository.

3.1 Task graph representation

A TG is divided into three major elements. The task graph element is the TG itself, which contains tasks and edges. A task element includes a task element for each task of the TG, while an edge element includes an edge element for each arc in the TG.

Each task has two main attributes: an unique identifier (*code*) and a task type (*type*), chosen among the 46 different types of tasks included in the E3S library [11]. Each edge has four main attributes: an unique identifier (*id*), an identifier of its source node (*src*), another of its target node (*tgt*) and an attribute representing the communication cost imposed (*cost*). Figure 1(a) shows the XML representation of a simple TG of E3S.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<task_graph>
  <tasks>
    <task code ="0" name="src"     type="45" />
    <task code ="1" name="text"    type="44" />
    <task code ="2" name="sink"    type="45" />
    <task code ="3" name="rotate" type="43"/>
    <task code ="4" name="dith"    type="42" />
  </tasks>
  <EDGES>
    <edge name="a0_0" from="0" to="1" cost="1000"/>
    <edge name="a0_1" from="0" to="3" cost="7070"/>
    <edge name="a0_2" from="3" to="4" cost="7070"/>
    <edge name="a0_3" from="4" to="2" cost="7070"/>
    <edge name="a0_4" from="1" to="2" cost="1000"/>
  </edges>
</task_graph>
```

(a) Example of a TG

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<repository>
  <ips>
    <ip procName="AMD_ElanSC520-133_MHz--square"
     price="33.0" taskTime="9e-06" taskPower="1.6"
     area="9.61E-6" taskName="Angle2Time Conversion"
     type="0" procID="0" id="0"
   />
   <ip procName="AMD_ElanSC520-133_MHz--square"
    price="33.0" taskTime="2.3e-05" taskPower="1.6"
    area="9.61E-6" taskName="Basic floating point"
    type="1" procID="0" id="1"
   />
   <ip procName="AMD_ElanSC520-133_MHz--square"
    price="33.0" taskTime="0.00049"taskPower="1.6"
    area="9.61E-6" taskName="Bit Manipulation"
    type="2" procID="0"id="2"
   />
    . . .
  </ips>
</repository>
```

(b) Example of an IP repository

Fig. 1. XML codes

3.2 Repository representation

The IP repository is divided into two major elements: the repository and the IPs elements. The repository is the IP repository itself. It is noteworthy to point out that the repository contains different non-general purpose embedded processors and each processor implements up to 46 different types of operations. Not all 46 different types of operations are available in all processors. Each type of operation available to be run in each processor is represented by an IP element. Each IP is identified by its unique identifier. It also includes other attributes such as taskType, taskName, taskPower, taskTime, processorID, processorName, processorWidth, processorHeight, processorClock, processorIdle, Area, Power and cost. The common element in TG and IP repository representations is the type attribute. Therefore, this element will be used to link an IP to a node. Figure 1(b) shows a simplified XML structure representing an IP repository. The repository contains IPs for digital signal processing, matrix operations, text processing and image manipulation.

4 The IP Assignment Problem

IP Assignment is the first step before mapping the application onto NoC [12]. The objective of IP Assignment is to select, from an IP library (IP repository), a set of IPs that exploit re-usability and optimize the implementation of a given application in terms of time, power and area requirements. During this step, no information about physical location of IPs onto NoC is given. The optimization process must be done based on TG and IP features only. The result of this step is a set of IPs that should maximize the NoC performance, *i.e.* minimize power consumption, hardware resources as well as the total time execution of the application. Recall that the result of this step is produced in the form of an ACG for the application's task graph, wherein each task has an IP associated with it. The dynamics of this assignment step is illustrated in figure 2. Note that the number of possible assignments is defined as in equation 1:

$$\#A = n_1 \times n_2 \times \cdots \times n_{m-1} \times n_m \tag{1}$$

wherein m represents the number of tasks used in the application and n_i the number of IPs that can be assigned to task i.

5 Assignment with DEMO Algorithm

In this section, we introduce the concept of Differential Evolution and its use for Multi-objective Optimization.

5.1 Differential evolution

Differential Evolution (DE), proposed by Storn and Price [13], is a simple and efficient Evolutionary Algorithm (EA). It was, initially, used to solve single-objective optimization problems [19]. DE is population-based global optimization algorithm, starting with a population of NP D-dimensional individuals. Each

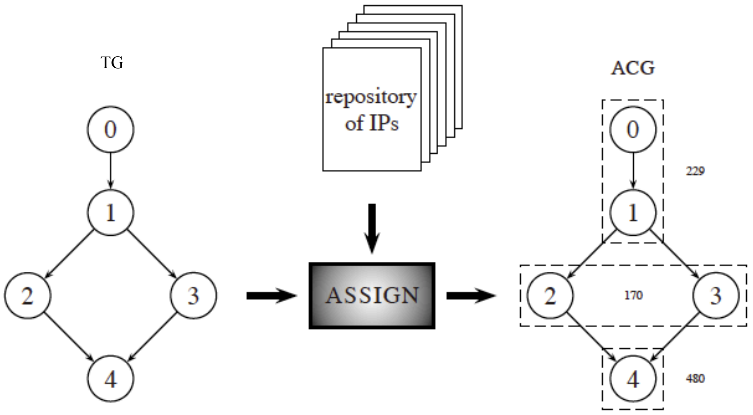


Fig. 2. Dynamics of the assignment process

individual encodes a candidate solution, $i.e\ X_{i,G} = \{X_{i,G}^{1}, ..., X_{i,G}^{D}\}$, $i = 1, ..., NP$, where G denotes the generation to which the population belongs [20]. The initial population is generated randomly from the entire search space. The main steps of the DE algorithm are as follows:

Mutation operation : The mutation operator allows altering or disrupting the population with the mutant vector $V_{i,G}$ for each individual $X_{i,G}$ in the population at the generation G. The mutation operator can be generated using a specific strategy. The most frequently used strategies are defined by equations 2 .. 6:

$$"DE/rand/1" :\ V_{i,G} = X_{r_1,G} + F.(X_{r_2,G} - X_{r_3,G}) \tag{2}$$

$$"DE/best/1" :\ V_{i,G} = X_{best,G} + F.(X_{r_1,G} - X_{r_2,G}) \tag{3}$$

$$"DE/best/2" :\ V_{i,G} = X_{best,G} + F.(X_{r_1,G} - X_{r_2,G}) + F.(X_{r_3,G} - X_{r_4,G}) \tag{4}$$

$$"DE/rand/2" :\ V_{i,G} = X_{r_1,G} + F.(X_{r_2,G} - X_{r_3,G}) + F.(X_{r_4,G} - X_{r_5,G}) \tag{5}$$

$$"DE/current - to - best/2" : V_{i,G} = X_{i,G} + F.(X_{best,G} - X_{r_1,G}) + F.(X_{r_2,G} - X_{r_3,G}) \tag{6}$$

where $V_{i,G}$ is the mutant vector to be produced. r_1, r_2, r_3, r_4, r_5 are integer constants generated randomly in the range of $[1, NP]$, which are different from the index i. $X_{best,G}$ is the best individual at generation G. F : scaling factor which is a real constant usually chosen in the range of $[0, 1]$. It controls the amplification of the difference variation.

Crossover operation : The crossover operation improves the diversity of the population and is applied after mutation phase. The crossover uses the mutation of the mutant vector $V_{i,G}$ to exchange its components with the target vector $X_{i,G}$ in order to form the trial vector $U_{i,G}$. The crossover operation is defined by equation 7:

$$U_{i,G}^{j} = \begin{cases} V_{i,G}^{j} & \text{if}(rand_j[0,1] \leq CR) or (j = j_{rand}) \\ X_{i,G}^{j} & \text{otherwise} \end{cases} \tag{7}$$

where $j = 1, 2, ..., D$. $rand_j$ is the jth evaluation of a uniform random number generator within $[0, 1]$ [17]. The crossover rate CR is a user-specified constant within the range $[0, 1]$. j_{rand} is a randomly chosen integer within the range $[1, D]$ [15].

Selection operation : In order to keep the population size constant over subsequent generations, a selection phase is performed. The trial vector is evaluated according to the objective function and compared with its corresponding target vector in the current generation. If the trial vector is better than target one, the trial vector will replace the target one otherwise the target vector will remain in the population. The selection operation is represented by equation 8:

$$X_{i,G+1} = \begin{cases} U_{i,G} & \text{if} f(U_{i,G}) \leq f(X_{i,G}) \\ X_{i,G} & \text{otherwise} \end{cases} \quad (8)$$

The three steps mutation, crossover and selection are repeated for each generation until a termination criterion.

5.2 Differential evolution for Multiobjective Optimization

In single-objective optimization, the decision in selection operation is straightforward, where the candidate just replaces the parent when the candidate is better than the parent. However, in multi-objective problems, we could use Pareto concept to deal with multiple objectives in order to select the best solution in the selection mechanism.

The DE algorithm was extended to solve multi-objective optimization problems due to its great robustness and fast convergence [20]. Among the multi-objective optimization algorithms based on DE, we find Differential Evolution for Multi-objective Optimization (DEMO) proposed by Robic and Filipic [19]. This algorithm uses the following selection strategy. If the newly generated trial vector dominates the parent vector, then the trial vector replaces the parent one. If the parent dominates the trial, the trial vector is discarded. Otherwise, when the trial and the parent vectors are not related to each other, the trial vector is added to the current population for later sorting. Algorithm1 shows the main algorithm of DEMO.

Algorithm 1 The main steps of modified DEMO

```
Initialize the individuals of the population
Initialize best solutions in archive of leaders
iteration := 0
while iteration $<$ max_iteration do
    for each individuals do
        Generate a mutated vector using a mutation operation
        Generate a trial vector using crossover operation
        Evaluate the trial vector
        If the trial vector dominate the individual , the trial vector replace the individual
        If the individual dominates the trial vector , the trial vectors is discarded
        Otherwise , the trial vector is added in the population.
    end for
    Update the leaders archive
    iteration := iteration + 1
end while
Return result from the archive of leaders
```

6 Objective functions

Different objectives may be considered in the IP assignment problem. The objectives can be concurrent or collaborative [4]. Concurrent objectives should not be grouped, and considered separately during optimization process. Thus, the process can be treated as a multi-objective optimization.

The best solution for multi-objective optimization is the solution with the adequate trade-off between all concurrent objectives. In this paper, we adopt a multi-objective optimization minimizing three objectives: area, power consumption and execution time. These objectives must be computed. Thus, we need to provide an executable definition for each of them.

Regarding the computation of the area required by a given assignment solution, it is necessary to know the area of each of the selected processor or processing element (pe). As a processor can execute more than one task, we must visit each task of the TG and associate it with the selected processor.

The area is computed summing up all the processor's area. However, when the same processor is associated to several tasks of the TG, the area of that processor is added only once. Equation 9 defines how to compute the required area to use for a given assignment A:

$$Area(A) = \sum_{pe \in Proc(ACG_A)} area_{pe}, \tag{9}$$

wherein $Proc()$ yields the set of processors used in ACG for the evaluated assignment A and $area_{pe}$ is the area for processor pe in ACG [11].

Moreover, the computation of the power consumption is obtained simply adding up all the TG task's static powers, as defined in equation 10:

$$Power(A) = \sum_{t \in ACG_A; pe \in Proc(ACG_A)} power_i^{pe}, \tag{10}$$

wherein $power_i$ is the required power consumption to execute task i on a specified processor pe [4].

Furthermore, regarding the execution time imposed for a given application, it is necessary to visit all the tasks of its TG and schedule them into the own cycle. Thus, a scheduling algorithm ought to be applied, for which we opted for List Scheduling (LS).

The LS algorithm [21] chooses, sequentially, in each iteration the ”best” task from all appropriate tasks to place into the next step. The tasks are sorted according to some ”priority function”. When the number of tasks exceeds the number of resources, the remaining tasks are delayed. Equation 11 defines the computation of the execution time:

$$Time(A) = \sum_{s \in Steps(TG)} \max_{pe \in s; t \in s} (time_t^{pe}), \tag{11}$$

wherein $time_t^{pe}$ returns the time needed to execute task t by processor pe.

7 Results

The E3S benchmarks suite was used to evaluate the proposed algorithm. The suite contains the characteristics of 17 embedded processors. Each processor is characterized by area, execution time and power consumption of 46 different tasks. The suite contains also some applications that are usually executed by embedded systems, such as telecommunication, auto-industry, and network. The applications are given by their task graph.

To apply the DEMO algorithm, parameters F and CR have to be chosen, for which a thorough sensitivity analysis is carried out. For mutation constant F and for CR, the sensitivity was evaluated for different values, such as 0.1 , 0.2 , 0.3 , 0.4 , 0.5 , 0.6 , 0.7 ,0.8 and 0.9. During this analysis, the following parameters provide good results during the simulation: the initial population was set to 100; $CR = 0.1$; and $F = 0.5$. The maximum number of iterations carried out was set to 200.

Among the common applications executed in a embedded environment, as provided by E3S, we considered the applications tested in [4]. In order to find the best performing strategy for the DEMO algorithm, different strategies of DEMO were evaluated. The computations of different DEMO variants could be recognized by using different equations for mutation operator, viz., equations 2...6.

The DEMO algorithm variants were applied to the above-described test problems. All the algorithms were run for the benchmark applications. The charts of figure 3 allow a visual comparison of the power consumption for the assignments yield by the compared strategies, regarding best and average results. Also, the charts of figure 4 allow for a better visual comparison of the results regarding the time characteristic, considering the best and average quality assignments. Finally, the charts of figure 5 facilitate the comparison of the required hardware area that would be required by the assignments obtained by the compared strategies.

In order to compare the DEMO algorithms strategies, different performance measures were used to evaluate the efficiency of each strategy. We must at least try to determine three aspects[22] : (1) if the set of approximations of non-dominated solutions covers the entire extension of the true Pareto Front; (2) if it is close to the true Pareto Front; and (3) whether the solutions in the approximation set are well distributed and spaced from each other. There are two types of indicators: unary, to measure the three above-mentioned aspects, or binary, that can be used for a direct comparison of the performances of two Pareto fronts. Since, in our work, we do not have the true Pareto front, only the binary indicators were used. Among the binary indicators, we chose Convergence-Based Indicators.

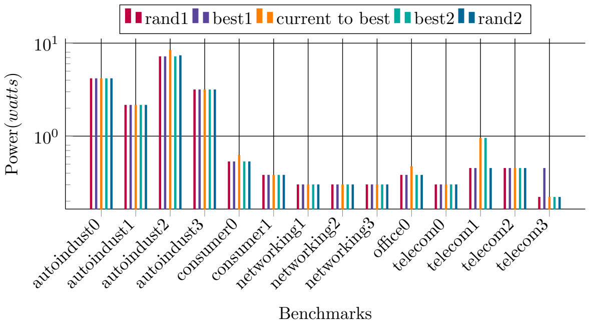


Fig. 3. Comparison of the minimum obtained assignments regarding power requirement

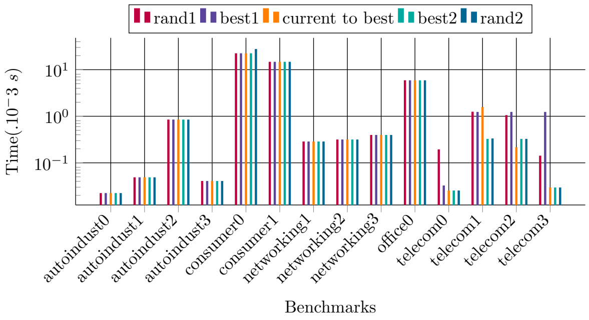


Fig. 4. Comparison of the minimum obtained assignments regarding time requirement

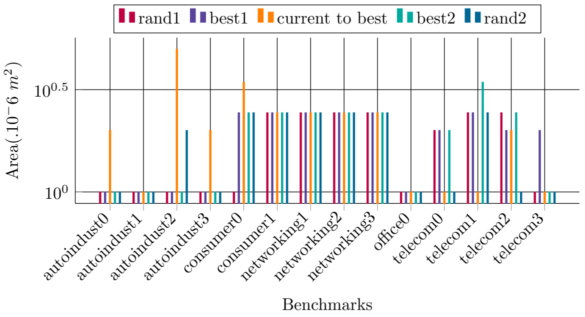


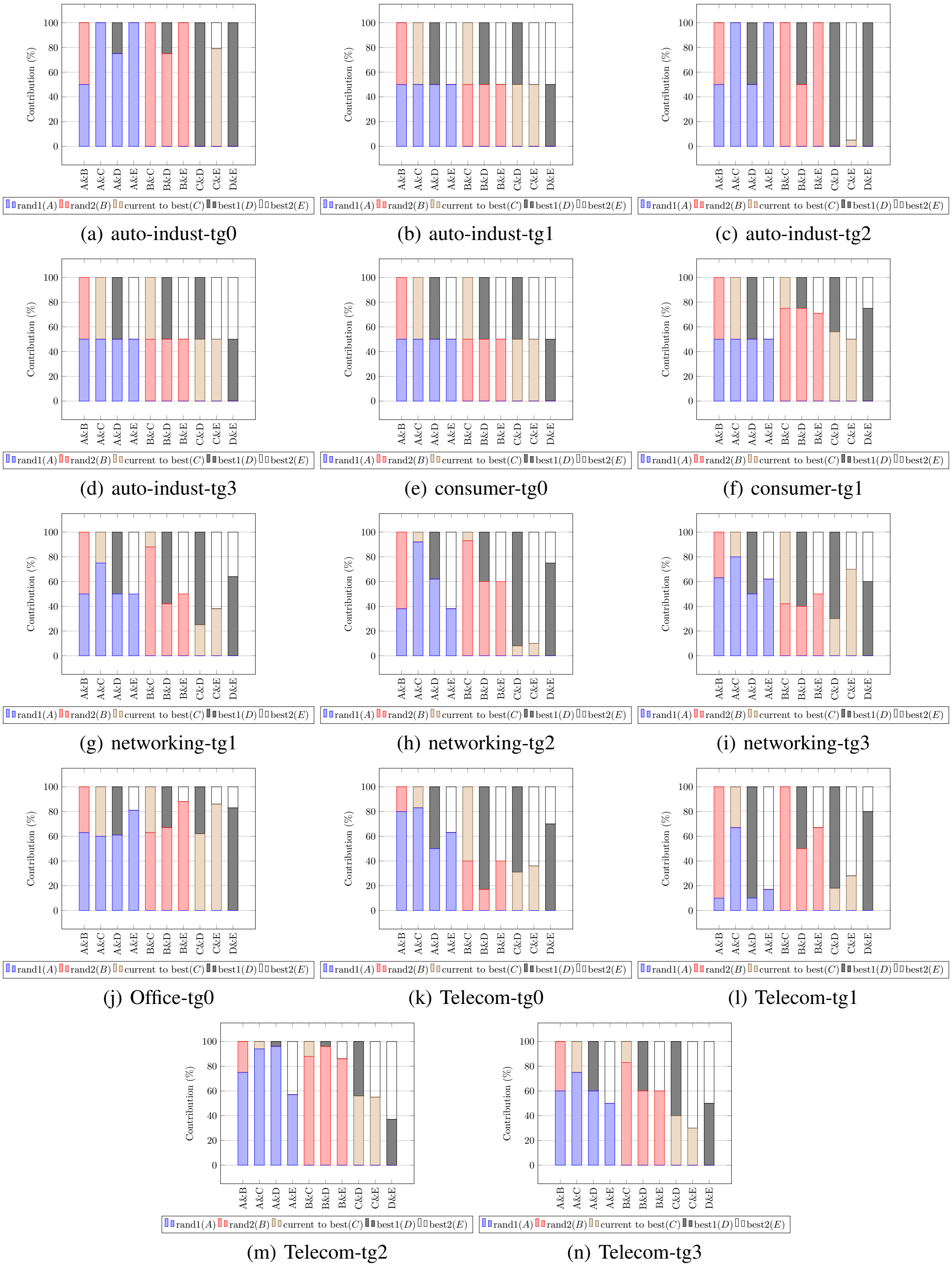
Fig. 5. Comparison of the minimum obtained assignments regarding area requirement

Contribution indicator [23] is one of the indicators used for convergence. The contribution of an approximation PO_1, relatively to another approximation PO_2, is the ratio of nondominated solutions produced by PO_1 in PO^*, which is the set of Pareto solutions of $PO_1 \cup PO_2$. PO is the set of solutions in $PO1 \cap PO2$. $W1$ (resp. W_2) is the set of solutions in PO_1 (resp. PO_2) that dominate some solutions of PO_2 (resp. PO_1). Let $L1$ (resp. $L2$) be the set of solutions in $PO1$ (resp. PO_2) that are dominated by some solutions of PO_2 (resp. PO_1). N_1 (resp. N_2) is the noncomparable solution of PO1 (resp. PO2):$Ni = PO_i/(PO \cup W_i \cup L_i)$. The contribution is calculated by equation 12:

$$Cont(PO1/PO2) = \frac{\frac{||PO||}{2} + ||W_1|| + ||N_1||}{||PO^*||}. \tag{12}$$

Note that $||PO^*|| = ||PO|| + ||W_1|| + ||N_1|| + ||W_2|| + ||N_2||$. For example, $Cont(A, B) = 0.7$ indicates that 70% of the solutions of the nondominated set of $A \cup B$ are provided by A, and then 30% provided by B. So, this value has to be greater than 0.5 to indicate that A is better than B in terms of convergence to the Pareto front. When $Cont(A, B) = 1$, all solutions of B are dominated by A.

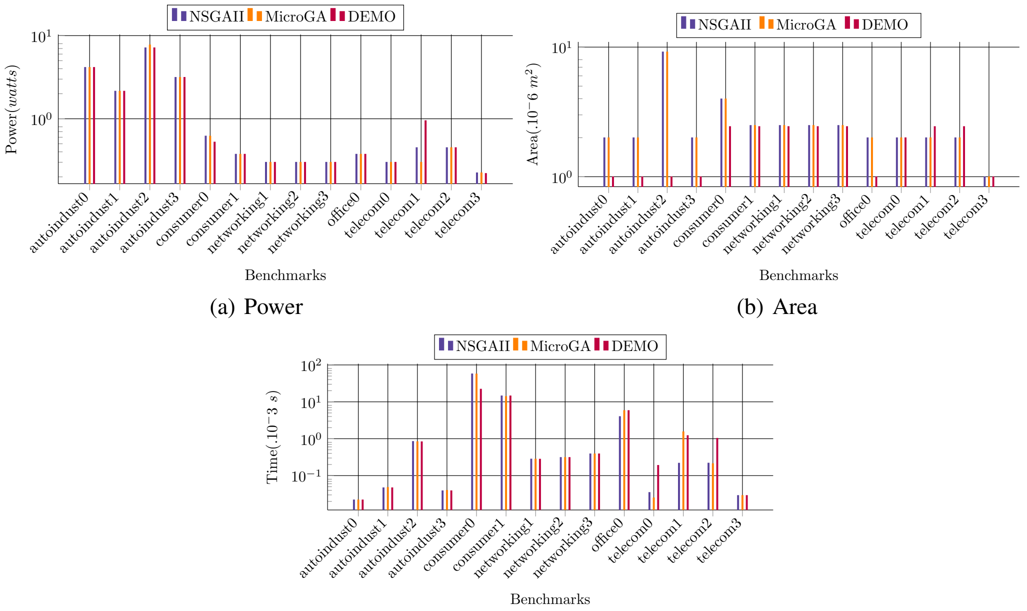
Different DEMO algorithms are denoted as A=DEMO/rand/1/; B=DEMO/rand/2/; C=DEMO/current-to-best/1/; D=DEMO/best/1/; E=DEMO/best/2/. In figure 6, the five DEMO variants are compared, showing the contribution between two strategies. The highest value indicates the best performance. It can be observed that the A strategy performs better than the others in most of applications. In networking-tg2,the B strategy is performes better than others. In networking-tg1, networking-tg3 and consumer-tg0, all strategies are equals.



(a) auto-indust-tg0 (b) auto-indust-tg1 (c) auto-indust-tg2 (d) auto-indust-tg3 (e) consumer-tg0 (f) consumer-tg1 (g) networking-tg1 (h) networking-tg2 (i) networking-tg3 (j) Office-tg0 (k) Telecom-tg0 (l) Telecom-tg1 (m) Telecom-tg2 (n) Telecom-tg3

Fig. 6. Contribution Value

In order to consolidate the proposed approach, we compared the obtained results to those reported in [4], where the NSGA-II and MicroGA were exploited to yield assignments. It is noteworthy to emphasize that the same objective function was used in the compared works. As shown in figure 7, it becomes clear that our approach with DEMO algorithm provides better results than NSGA-II and MicroGA, whatever the implemented DEMO strategy. However, considering the time objective, some benchmarks are best served with NSGAII and MicroGA.



(a) Power

(b) Area

(c) Time

Fig. 7. IP Assignment Comparison using NSGA II and MicroGA and DEMO

8 Conclusions

The IP assignment step is an NP-hard combinatorial problem in NoC design. In this paper, we proposed a multi-objective algorithm based on Deferential Evolution to help NoC designers to select a set of appropriate IP blocks from the IP repository. We used the TG and ACG structure, presented in [11], and we adopted the E3S benchmark as the IPs benchmark. We used the same objectives used in [4] and compared five DEMO strategies. The efficiency was compared with convergence metric [23] and we found that DEMO with A strategy performed best among all other strategies and also with relation to the results presented in [4]. For future work, we intend to extend the measures to calculate the divergence metric and also to use the same algorithm to tackle the mapping step.

References

1. Jingcao Hu and Radu Marculescu. Energy-aware mapping for tilebased NoC architectures under performance constraints. In ASPDAC: Proceedings of the 2003 conference on Asia South Pacific design automation, pages 233–239, New York, NY, USA, 2003. ACM.
2. Umit Y. Ogras, Jingcao Hu, and Radu Marculescu. Key research problems in NoC design: a holistic perspective. In Proceedings of the 3rd IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis, CODES+ISSS 2005, Jersey City, NJ,USA, September 19–21, 2005, pages 69–74. ACM, 2005.
3. M. R. Garey and D. S. Johnson. Computers and intractability; a guide to the theory of NP-completeness. Freeman and Company, 1979.
4. Marcus Vinícius Carvalho da Silva, Nadia Nedjah, Luiza de Macedo Mourelle:Evolutionary IP assignment for efficient NoC-based system design using multi-bjective optimization. IEEE Congress on Evolutionary Computation 2009: 2257–2264

5. Robert P. Dick, David L. Rhodes, and Wayne Wolf. TGFF: Task Graphs For Free. In Proceedings of the 6th International Workshop on Hardware/Software Co-esign, pages 97–101, Seattle, Washington, USA, March 1998. IEEE Computer Society.
6. Wenbiao Zhou, Yan Zhang and Zhigang Mao. Pareto based multiobjective mapping IP cores onto NoC architectures. In APCCAS, pages 331–334. IEEE, 2006.
7. Rabindra Ku. Jena and Gopal Ku. Sharma. A multi-objective evolutionary algorithm based optimization model for network-on-chip synthesis. In ITNG, pages 977–982. IEEE Computer Society, 2007.
8. Ciprian Radu, Optimized Algorithms for Network-on-Chip Application Mapping ANoC, PhD Thesis, University of Sibiu, engineering Faculty Computer Engineering Department, 2011.
9. Ciprian Radu, M.D. Shahriar Mahbub, Lucian Vintan: Developing Domain-Knowledge Evolutionary Algorithms for Network-on-Chip Application Mapping. Microprocessors and Microsystems - Embedded Hardware Design 37(1): 65–78 (2013).
10. The World Wide Web Consortium. World Wide Web Consortium(W3C): `http://www.w3.org`, 2008.
11. Robert P. Dick. Embedded System Synthesis Benchmarks Suite (E3S):`http://ziyang.eecs.northwestern.edu/dickrp/e3s/`.
12. Nadia Nedjah, Marcus Vincius Carvalho da Silva, Luiza de Macedo Mourelle: Preference-based multi-objective evolutionary algorithms for power-aware application mapping on NoC platforms. Expert Syst. Appl. 39(3):2771–2782 (2012)
13. STORN, Rainer et PRICE, Kenneth. Differential evolutiona simple and efficient heuristic for global optimization over continuous spaces. Journal of global optimization, 1997, vol. 11, no 4, p. 341-359.
14. DAS, Swagatam et SUGANTHAN, Ponnuthurai Nagaratnam. Differential evolution: A survey of the state-of-the-art. IEEE transactions on evolutionary computation, 2011, vol. 15, no 1, p. 4-31.
15. MALLIPEDDI, Rammohan, SUGANTHAN, Ponnuthurai N., PAN, Quan-Ke, et al. Differential evolution algorithm with ensemble of parameters and mutation strategies. Applied soft computing, 2011, vol. 11, no 2, p. 1679-1696.
16. FAN, Qinqin et YAN, Xuefeng. Multi-objective modified differential evolution algorithm with archive-base mutation for solving multi-objective *p*-xylene oxidation process. Journal of Intelligent Manufacturing, 2018, vol. 29, no 1, p. 35-49.
17. REDDY, M. Janga et KUMAR, D. Nagesh. Multiobjective differential evolution with application to reservoir system optimization. Journal of Computing in Civil Engineering, 2007, vol. 21, no 2, p. 136-146.
18. MEZURA-MONTES, Efrn, REYES-SIERRA, Margarita, et COELLO, Carlos A. Coello. Multi-objective optimization using differential evolution: a survey of the state-of-the-art. In : Advances in differential evolution. Springer, Berlin, Heidelberg, 2008. p. 173-196.
19. ROBIC, Tea et FILIPIC, Bogdan. Differential evolution for multiobjective optimization. In : International Conference on Evolutionary Multi-Criterion Optimization. Springer, Berlin, Heidelberg, 2005. p. 520-533.
20. M. Ali, P Siarry and M. Pant, An efficient differential evolution based algorithm for solving multi-objective optimization problems, *European journal of operational research*, vol. 217, no. 2, p. 404–416, 2012.
21. Pangrle BM, Gajski DD. Design tools for intelligent silicon compilation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 1987, vol. 6, no 6, p. 1098-112.
22. AZEVEDO, Carlos RB et ARAJO, Aluizio FR. Correlation between diversity and hypervolume in evolutionary multiobjective optimization. In : Evolutionary Computation (CEC), 2011 IEEE Congress p. 2743-2750.
23. TALBI, El-Ghazali. Metaheuristics: from design to implementation. John Wiley and Sons, 2009.

A modified fixed point method for biochemical transport

M.R. AMATTOUCH[1] and H. Belhadj[1]

[1] University of Abdelmalek Essaadi, Faculty of sciences and techniques, department of mathematics,
BP.416, Tangier, Morocco
amattouch36@gmail.com
[2] hassan.belhadj@gmail.com

Abstract: This work is devoted to a modified fixed point method applied to the bio-chemical transport equation. To have a good accuracy for the solution we treat, we apply an implicit scheme to this equation and use a modified fixed point technique to linearize the problem of transport equation with a generalized nonlinear reaction and diffusion equation. Next, we apply this methods in particular to the the dynamical system of a bio-chemical process.
Eventually, we accelerate these algorithms by the optimized domain decomposition methods.
Several test-cases of analytical problems illustrate this approach and show the efficiency of the proposed new method.

Key words: Bio-chemical transport, Modified fixed point method, optimal control.

1 Introduction

Our general equation of bio-chemical transport equation, could be written as:

$$\begin{cases} \frac{\partial u}{\partial t} + -div\,(D(u)\nabla u) + \nabla vu + F(u) = f(x,y) \;\; on\; \Omega \\ \qquad\qquad\qquad\qquad\qquad\qquad u = h \;\; on\; \partial\Omega \\ \qquad\qquad\qquad\qquad\qquad\qquad u = h_o \;\; on\; t = 0 \end{cases} \tag{1}$$

Where $u = (u_i)$ the concentration (dispersal) of reactant, F is the reaction model and D is the diffusion flux. The lists of bio-chemical model is too wide to cite in this paper, but our boundary equation (1), generalizes a lot of bi-chemicals models (with second order derivative). As concerned biochemical models, we cite the Fisher equation (for tumors model [5, 15]), Murray Model [6], Keller and Segel(for Chemotaxis process [7]) and FitzHugh-Nagumo (for the simulation of electric propagation in nerves [8]),...
The use of Hopf biffurcation or continuation methods are insufficient for modeling in general these type biochemical dynamic and solving these models cost heavy time. Then, the use of accurate numerical methods is necessary to solve the dynamics. Notice, the use of the Newton type methods is ineffective for solving nonlinear equation of dynamics because the dimension of variables in these equations is large (more than 3 variables). Also, the use of explicit scheme to solve these equations is not accurate, stable and efficient in general . So In order to make ours bio-chemicals simulations fast, stable and efficient, we apply in this paper, the modified fixed point (we propose in [1, 2,4]) to solve problem (1) type. Modifying the treated equations someway to make the global energy of the system contracting at each step of time, we guarantee that our method converge quickly to the stationary solution and the time step size is realistic compared to an explicit method (because of the CFL condition impose a very small time step size for explicit scheme). Also, modifying the equations of dynamics give stable solutions: The use of finite element or finite volume discretization is sensitive to the anisotropic and convective equations: modifying equations by q relaxation procedure solves this issue and we show the weak formulation of problem (1), has one and unique solution.
In the first part of this paper, we will present the modified fixed point method applied to our equation (1). In the second part we apply this method to the particular case of a dynamical reaction equation without a diffusion or convection term. We are interested in this section, because the equations type we treat, model a lot of bio-chemical process (the Michaleis-Menton for Enzyme kinetics [9, 16], Hill Kinetics [10], Goldbeter-Koshland for switch phenomena [11], Tyson Model for cell cycles [12], Burgos Fang Model [13] ...). The equations we treat in this paper in general are unstable and hard to solve by classical methods, so we apply the modified fixed point to make the solutions robust and stables.
Finally, we give some numerical results and implementation that prove the efficiency of the method.

2 Nonlinear reaction diffusion equation

By a full implicit scheme, equation (1) becomes

$$\begin{cases} cu - div\, D(u)\nabla u + vu = f(x,y) \;\; on\ \Omega \\ \qquad\qquad\qquad\qquad u = h \;\; on\ \partial\Omega \end{cases} \tag{2}$$

The fixed point method is constructed successively as:

$$\begin{cases} cu_{n+1} - div\, D(u_n)\nabla u_{n+1} + \nabla v u_{n+1} + F(u_{n+1}) = f(x,y) \;\; on\ \Omega \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad u_{n+1} = o \;\; on\ \partial\Omega \end{cases} \tag{3}$$

Where u_0 is an initial function value.
Let

$$V = \{u \in H(\Omega)/\ D(u) \in L^1(\Omega)\ and\ \|u\|_H < M\}$$

H is the specific Sobolev space adopted for problem (1) (we take $H(\Omega) = H_0^1(\Omega)$ for a dirichlet condition on the boundary and $H(\Omega) = H^1(\Omega)$ for a Neuman condition) and $\|u\|_H$ is the norm on the Sobolev space H ($\|u\|_H = \|\nabla u\|$ in the case of dirichlet condition and $\|u\|_H = \|u\| + \|\nabla u\|$ in the case of neumann condition). M is a positive constant that we choose to make the solution of the variational formula of equation (2) to be bounded (for physical reason). The energy of equation (2) is:

$$E(u) = \int_\Omega cu\, u + \int_\Omega u\, \nabla v u + \frac{1}{2}\int_\Omega D(u)\nabla u\, \nabla u - \int_\Omega f u$$

Equation (2) is equivalent to the global minimization of the Energy E on the space H with some constraints. We can prove that E is a K contraction on V for a small number M (see [1,2] for prove and explanation in the case of nonlinear diffusion). This method is local and could diverge for an initial sequence departure is $u_0 = h_0$. the same thing could be said if we apply a Newton type method. As a solution to this problem we make some modifications to the equation to have successive sequences that are not distant from the initial value $u_0 = h_0$. As in [2] where, we proposed a modified fixed point to a just a semi linear equation, and in [1] where, we applied a modified fixed point to a nonlinear equation, we generalize this method to our equation by mean of solving the following iterative equations:

$$\begin{cases} cu_{n+1} - div((D(u_n) + r(u_n)), \nabla u_n)\nabla u_{n+1}) + (u_{n+1}) = f(x,y) - div(r(u_n), \nabla u_n)\nabla u_n) + K(u_n) \;\; on\ \Omega \\ \qquad\qquad\qquad\qquad\qquad\qquad\qquad u_{n+1} = o \;\; on\ \partial\Omega \end{cases} \tag{4}$$

the function r and K are selected such a way that the valuational energy of this equation is a K contraction where, K is very small for reasonable choice of bound M (Generally M Must be less than $\|u_0\|_H$). This small K make the convergence of the method fast and reduce the number of successive iterations in time compared to Newton or Fixed method ones. Also, the convergence to the solution is stable because the energy is a contraction (there is one and only one solution for the problem and the error is controlled).

3 Dynamical reaction model

In this section, we treat a particular case of equation (1): We eliminate the convective and diffusive term and take into account a non linear reaction: Without the transport term in equation, we obtain a dynamical equation:

$$\frac{dy}{dt} = F(y), \quad y(0) = u_0 \tag{5}$$

F is non linear function that could have many roots, which make the equation unstable and accurate using classical model from initial value u_0.
An implicit scheme of this equation could be writing as:

$$y_{n+1} = F(y_{n+1}) + Ky_n \;\; y_0 = u_0 \tag{6}$$

We do the same job of section 2 and apply a modified fixed point method to the problem(instead of a continuation method that make a lot of time to process):

$$y_{n+1} = F(y_n) + c_n y_{n+1} + (K - c_n) y_n \tag{7}$$

c_n is choose such a way that the energy of y, should be a contraction.
We can take for example::

$$c_n = \nabla F(y_n) + \frac{1}{2} y_n . HessF(y_n)$$

Hess is the matrix hessian for the function F.

4 Numerical simulation

We treat first the simulation of equation (1) where Ω is a squared domain, we take different values for the diffusion D and the convection v, the functions f, h and h_0 are given by taking an exact solution u_{exact} to the problem (1) on the square Ω. We implemented the modified fixed point method to several academic solution u_{exact}. We take in this paper in the case of function of two variable:

$$u_{exact}(x, y, t) = ((2x - y)e^{-t(x^2+y^2+} + xsin(\pi y), (2x + y)e^{-t(x^2+y^2)} + ycos(\pi x) + 1)$$

The next figures show the error between the approximate solution by a finite element method and the analytical solution of equations (4): $u_{exact}(., t)$ in the given step t for different value D, F, v=(a,b), Δt, the step t and the mesh h of the finite element.

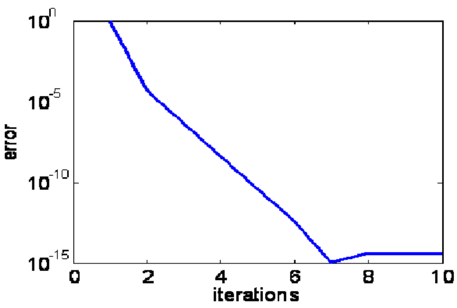


Fig. 1. case1.

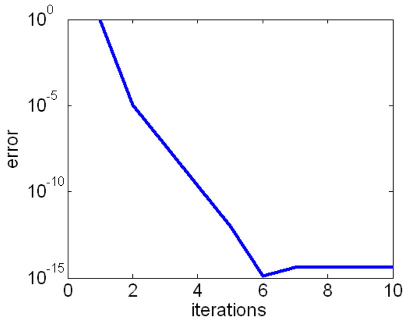


Fig. 2. case2.

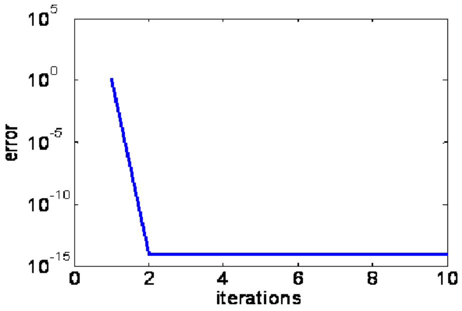


Fig. 3. case3

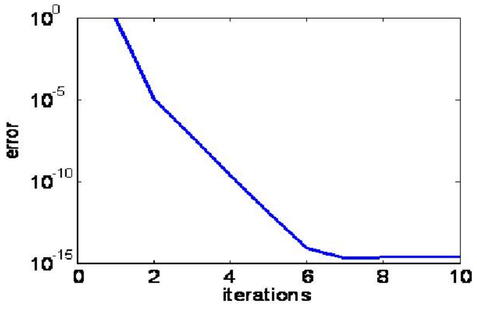


Fig. 4. case4

figures 1,2,3,4 showing the spatial error at a given time t in ms between the approximate solution by the fixed point method and the given exact solution of the problem (1) in terms of the given iterations

case1: Δt=0.1, t=0.1,D=I,a=b=0 h=0,025.$F(u, v) = (u * v - v, u)^t$.
case2: Δt=0.1, t=0.5, D=0,1I,a=-2,b=1 and h=0,001.$F(u) = a/(b + \|u\|)(1, 1)^t$
case3: Δt=0.1,t=10,$D = [1, 1 - e^{-(x+y)}; -1, 2]$,a=-1,b=0 and h=0,001.$F(u) = (u, u^3 - vu)^t$,
case4: Δt=0.5,t=1, D=0.5I, a=2, b=-1,5 and h=0,001.$F(u) = (\frac{u}{1+u^2}, \frac{v}{1+u^2})^t$.
h is the mesh grid and I is the matrix identity.

These figures shows that the modified fixed point method converge quickly to the solution and give a good accuracy. To reduce time computation due to the implicit scheme, we have combined this method an optimized wave domain decomposition method to accelerate the algorithm.
For the dynamical system (Equation (6))we give the simulation of a reaction with four reactant the bio-chemical equation (Equation 5).

$$X_1 + E \rightleftharpoons X_2 + \frac{1}{2}X_3 \rightleftharpoons X_4 + F$$

We take for F a polynomial function as for the Michaleis-Menton model for enzymes ([16]). We take for the initial value $[X_1] = 1$ and $[X_i] = 0$ for i=2,3,4. the next figure show the result of concentration species by solving the equation using the proposed modified fixed method in section 3. these results are close to a simulation with a benchmark of biochemical simulation (PyMol).

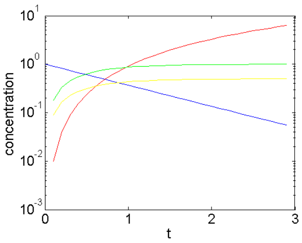


Fig. 5.

5 Conclusion

We have applied a modified fixed point method to resolve the general nonlinear transport equation, then we applied the method to a dynamical system. We assume that we can prove by theory the fast convergence of the method applied to this equation.
Several test-cases show the efficiency of the modified Fixed point method.
As a perspective of this article about modified fixed point method we can treat:

- Applying the method to the Groundwater equations
- Applying the Fick and Darcy equations
- Comparing results with realistic experiments.

References

1. M.R. Amattouch, H. Belhadj, *Combined Optimized Domain Decomposition Method and a Modified Fixed Point Method for Non Linear Diffusion Equation*, Applied Mathematics and Information Sciences, 11, No. 1, 201-207 (2017).
2. M.R. Amattouch, N. Nagid, H. Belhadj, *Optimized Domain Decomposition Method for Non Linear Reaction Advection Diffusion Equation*, European Scientific Journal , Vol 12, No 26 (2016).
3. M.R. Amattouch, N. Nagid, H. Belhadj, *a new splitting method for the Navier Stokes equation* , Journal of space exploration, Vol 2, 24 august 2017.
4. M.R. Amattouch, N. Nagid, H. Belhadj, *A modified fixed point method for The Perona Malik equation*,Journal of Mathematics and System Science 7, 175-185, september 2017
5. Fisher R.A, *The wave of advance of adventage genes.*,Ann. Eugenics, Vol. 7, pp. 353-369, 1937

6. Murray J.D, *Mathematical Biology.*, Berlin: Springer. 1993.
7. T. Hillen , K. J. Painter , *user's guide to PDE models for chemotaxis*,Journal of Math. Biol.(2009) 58,183-217
8. A. M. Turing, *he chemical basis of morphogenesis*, Philosophical Transactions of the Royal Society of London , B 737, 1953, pp.37-72.
9. R. FitzHugh, *pulses and physiological states in theoretical models of nerve membrane*, Biophys. J. 1 (1961) 445-466.
10. Chang, Raymond. *Physical Chemistry for the Biosciences.* Sansalito, CA: University Science, 2005. Page 363-371.
11. A. V. Hill, *The possible effects of the aggregation of the molecules of homoglobin on its dissociation curves*, J. Physiol., 40,iv-vii, 1910
12. Goldbeter A, Koshland DE, *An Amplified Sensitivity Arising from Covalent Modification in Biological-Systems.*,Proc Na tl Acad Sci USA 78: 6840-6844, 1981
13. Tyson JJ, *Modeling the cell division cycle: cdc2 and cyclin interactions*,Proc. Natl. Acad. Sci. U.S.A. 1991;88:7328-7332.
14. Y Fang, GT Yeh, WD Burgos, *A general paradigm to model reaction?based biogeochemical processes in batch systems*Water Resources Research, 2003
15. Benzekry, S., *Modeling and mathematical analysis of anti-cancer therapies for metastatic cancers*,PhD thesis University of Aix-Marseille (2011).
16. J. Nagumo, S. Arimoto, S. Yoshizawa, *An active pulse transmission line simulating nerve axon*, Proc. IRE 50 (1962) 2061-2070.

JOURNAL OF THE OPERATIONAL RESEARCH SOCIETY
JORS
Thomas Archibald, Jonathan Crook

Journal of the Operational Research Society

ISSN: 0160-5682 (Print) 1476-9360 (Online) Journal homepage: http://www.tandfonline.com/loi/tjor20

Multi-objective optimization of the integrated problem of location assignment and straddle carrier scheduling in maritime container terminal at import

Hamdi Dkhil, Adnan Yassine & Habib Chabchoub

To cite this article: Hamdi Dkhil, Adnan Yassine & Habib Chabchoub (2018) Multi-objective optimization of the integrated problem of location assignment and straddle carrier scheduling in maritime container terminal at import, Journal of the Operational Research Society, 69:2, 247-269, DOI: 10.1057/s41274-017-0184-9

To link to this article: https://doi.org/10.1057/s41274-017-0184-9

Published online: 16 Jan 2018.

JOURNAL OF THE OPERATIONAL RESEARCH SOCIETY, 2018
VOL. 69, NO. 2, 247–269
https://doi.org/10.1057/s41274-017-0184-9

Multi-objective optimization of the integrated problem of location assignment and straddle carrier scheduling in maritime container terminal at import

Hamdi Dkhil[a], Adnan Yassine[a,b] and Habib Chabchoub[c]

[a]LMAH, FR CNRS 3335, Normandie Université, ULH, Le Havre, France; [b]Institut Supérieur d'Études Logistiques (ISEL), Normandie Université, ULH, Le Havre, France; [c]International School of Business Sfax, Sfax, Tunisia

ABSTRACT

Maritime terminals need more efficiency in their handling operations due to the phenomenal evolution of world container traffic, and to the increase of the container ship capacity. In this work, we propose a new integrated modeling considering the optimization of maritime container terminals using straddle carriers. The problem is considered at import. We study a combination between two known problems, the first is the storage location assignment problem, and the second is the straddle carrier scheduling problem. This approach, which combines between two chronologically successive problems, leads to the use of multi-objective optimization. In fact, we study the multi-objective integrated problem of location assignment and Straddle carrier Scheduling (IPLASS) in maritime container terminal at import. We prove that the problem is NP-Complete. The objective is to minimize the operating cost which we evaluate according to eight components: the date of last task called makespan, the total vehicle operating time, the total storage bay occupation time, the number of vehicles used, the number of storage bays used, the number of storage locations used, and two different costs of storage location assignment. The location assignment costs are evaluated in order to facilitate the containers transfer for deliveries. We assume that the operating cost is a function of these components and that the influence of each component is variable and dependent on different parameters. These parameters are essentially: the number of quays in the terminal, the straddle carrier traffic layout, the number of container ships to serve in the terminal, the influence of concurrent operations in the terminal, the storage space configuration, the number of free storage bays, the number of free straddle carriers, the number of free quay cranes, the mobility of quay cranes, etc. To solve IPLASS efficiently, we propose an adapted multi-objective Tabu Search algorithm. Lower-bound evaluations are introduced to perform approximation of Pareto Front. To explore efficiently the non-convex Pareto Front Region, we evaluate also a maximized distance adapted to the set of objectives. Indicators of efficiency are developed to propose distinguished solutions to operator. *2D*-projections of approximated Pareto Frontier are given to more understand the efficiency of proposed solutions.

ARTICLE HISTORY
Received 20 May 2014
Accepted 18 January 2017

KEYWORDS
Maritime container terminal; straddle carriers; containers; location assignment; scheduling; multi-objective optimization; NP-complete problem; meta-heuristic; Tabu Search algorithm

1. Introduction

The world container port traffic grew from 300 million TEUs (Twenty Equivalent Units) in 2003 to more than 651 million TEUs in 2013 and the world container fleet capacity grew from 1.7 million TEUs in 1990 to 16 million TEUs in 2008 (Figure 1).

If we compare the number of orders for container ships to the number of container ship deliveries in 2007 (see Table 1), we can conclude that the world container traffic requires more and more ships in particular ships with a capacity of more than 10000 TEUs. Notice that in 2007, 134 of these container ships were recorded in order books and only 7 were delivered. In the same year, the orders of container ships that can carry more than 7500 TEUs represented 34% of the container ships ordered.

At import, maritime container terminal (MCT) guarantees especially three tasks: discharging containers fromships, stacking containers in storage space and finally delivering containers to shippers and consignees. At export, MCT guarantees the same tasks but in the opposite order. These three tasks are composed by other tasks which represent important optimization problems: berth allocation, yard planning, stowage planning, quay crane (QC) scheduling, vehicle scheduling and routing (straddle carriers, Automated Guided Vehicle, Automated Lifting Vehicle etc.), yard crane scheduling, logistics planning of operations. Logistic planning provides an efficient coordination between the different equipment and decisions at MCT.

Many kinds of handling systems are used at maritime terminals. In this paper, we study the case of CHS

CONTACT Hamdi Dkhil hamdi.dkhil@univ-lehavre.fr

This paper has been re-typeset by Taylor & Francis from the manuscript originally provided to the previous publisher.
Supplemental data for this article can be accessed at https://doi.org/10.1057/s41274-017-0184-9.

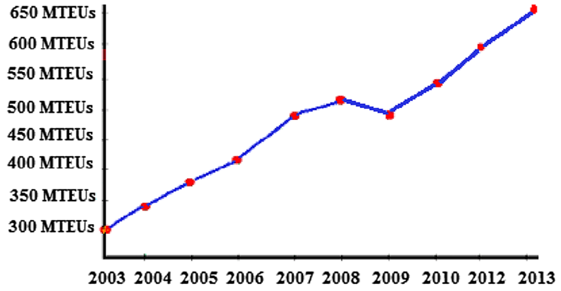


Figure 1. World container port traffic between 2003 and 2013. *M TEUs* 1 million twenty foots equivalent units.
Data source: www.worldbank.com.

Table 1. Container ships commands and deliveries in 2007.

Ship capacity (TEUs)	Commands Number of ships/ total capacity (TEUs)	Deliveries Number of ships/ total capacity (TEUs)
<10000	134/1659092	7/96124
7500/10000	78/673778	34/300516
6000/7500	39/257014	27/181630
5250/6000	9/49950	5/29112
4000/5250	130/576015	65/305169
3000/3999	31/108374	25/88670
2000/3000	63/160465	43/113481
1000/2000	126/177116	115/161241
<1000	62/51359	13/11732
All ship	606/3637957	400/1362881

Data source: French center of maritime studies—ministry of development and transport.

using straddle carriers to transfer containers between the quay and the storage space. Different blocks compose the storage space where every block is a set of storage bays and each bay is composed by different storage locations (Figure 2). The maximal capacity of storage locations is generally equal to three or four containers vertically stored. Using straddle carriers in MCT, there is no yard crane (YC), the vehicles (straddle carriers) directly access to the storage bays.

In this paper, we study an integrated problem which combines between the allocation and scheduling of the equipment and the location assignment in MCT using straddle carriers at import. These two problems are generally treated separately. This combination improves the productivity of the handling system due to a better theoretical optimality, and it promotes the use of multi-objective optimization (MOO).

The only study of the integrated problem of storage space allocation and vehicle scheduling, in the general context of container terminals, was the study by Bish, Leong, Li, Ng, and Simchi-Levi (2001) which treats the problem for automated container terminals (terminals with handling system using Automated Guided Vehicles denoted by AGV). Throughout this research, the vehicle schedule and location assignment are optimized in order to minimize one objective function which is the handling time. However, the waiting times in bay entry (AGVwaits for stacking crane in bay entry), which is a crucial constraint of the real problem, is not considered. In other studies, the optimization of the storage location assignment in container terminal considers total vehicle routing distance. But, vehicle scheduling, waiting time in bay entries, as well as the interaction between the different equipment and other parameters, are not considered.

In our study, we consider the multi-objective aspect of the problem with eight realistic objectives to be minimized. It is a new and efficient approach considering the state of art. We treat with the following objectives: the makespan (date of last task or operating time), the number of straddle carriers used, the sum of straddle carrier operating times, the sum of storage bay occupation times, the number of storage bays used, the number of storage locations used and two location costs.

We propose a new and efficient MOO approach applied to the Integrated Problem of Location Assignment and Straddle Carrier Scheduling (IPLASS) in MCT at import. First, we present the problem, then we introduce its mathematical modeling and finally we discuss a new multi-objective Tabu Search algorithm (MOTSA) adapted to solve efficiently combinatorial multi-objective optimization problems (MOOP) generally and IPLASS particularly. Note that we demonstrate the NP-Completeness of IPLASS in annex 1 of the online supplementary material.

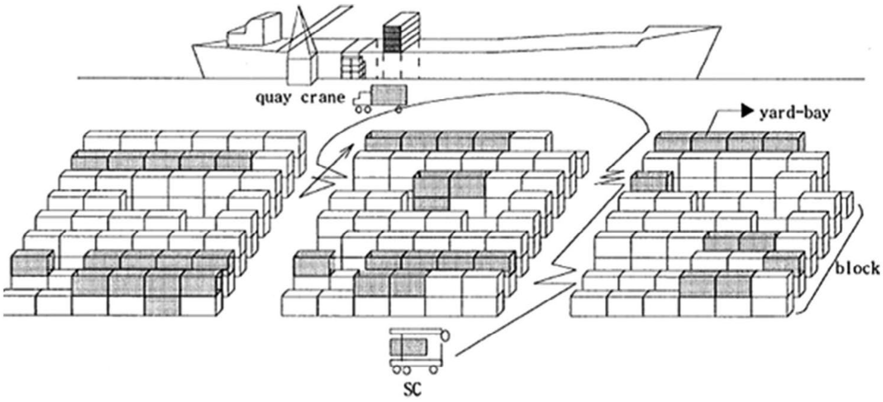


Figure 2. Storage block and storage bays. Source: http://pubs.sciepub.com/ijefm/2/1/5/.

2. Literature review

In this part, we present different reviews of literature treating optimization problems at MCT managed by straddle carriers, and MOO in general context.

2.1. Mono-objective optimization of MCT handling system

In this section, we describe the most important studies treating optimization of handling system of straddle carriers in MCT.

Kim (1997) evaluated the number of re-handles in container yards. The author discussed a set of equations to estimate this number. Kim and Kim (1999a) developed a beam search algorithm for the straddle carrier routing problem at export. The approach comprises the container location problem and the carrier routing problem. The authors treated only one objective, minimizing the total travel distance of straddle carriers in the yard. In the same year, Kim and Kim (1999b) developed a segregating space allocation modeling for import container inventories in port container terminals. The objective is to minimize the expected total number of re-handles. The authors discussed different procedures to solve the problem. Kim, Park, and Ryu (2000) discussed driving decision rules to solve the storage space allocation problem. They considered the goal of minimizing the number of relocation movements expected for the loading operation. The authors developed a dynamic programming model. To solve the real-time problem, they used a decision tree taking into account the optimal solutionsof the dynamic programming model. Meersman and Wagelmans (2001) discussed the integrated problem of QC–AGV–ASC (Quay Crane–Automated Guided Vehicle–Automated Stacking Crane) planning. He also proposed new modeling and developed branch and bound algorithm and beam search approach to solve the problem. In Nishimura, Imai, and Papadimitriou (2005), discussed the problem of yard trailer routing at a maritime container terminal. They developed a dynamic routing approach to solve the problem. Chen, Bostel, Dejax, Cai, and Xi (2007) presented a Tabu Search algorithm to solve the integrated problem of container-handling systems in a maritime terminal. The authors presented and discussed the problem as a hybrid flowshop scheduling problem. The proposed approach minimizes only one objective which is the total distance of rooting task. All these studies have not considered the multi-objective aspect of optimization at MCT.

2.2. Continuous multi-objective optimization

As the name suggests, MOO considers different goals in only one global problem. In industry and logistics, the first resolutions of MOOP have transformed these problems to single-objective optimization Problems in order to solve them. However, there are many differences between these two cases. In fact, decision-making for MOOP needs a new generation of multi-objective algorithms (MOA). After resolution of MOOP, the result is generally a set of solutions and the operators have to choose one of them. To select one efficient solution, different methods are proposed. We describe these methods as multi-objective election methods.

The most used MOA in the literature are meta-heuristic algorithms. Genetic Algorithm (GA), Evolution Strategies, Simulated Annealing (SA) and Tabu Search (TS) are particularly used.

The most used meta-heuristic for MOOP is GA. Deb, Pratap, Agarwal, and Meyarivan (2002) developed multi-objective GA named NSGA-II.

NSGA-II is a non-dominated sorting genetic multi-objective evolutionary algorithm. Authors compared NSGA-II to different effective variants of GA. NSGA-II performed the other algorithms for nine test problems, it highlights three famous difficulties concerning multi-objective evolutionary approach: the $O(MN^3)$ computational complexity (where M is the number of objectives and N is the population size), the non-elitism and the determination of sharing parameters. Actually, the algorithm is a $O(MN^2)$ computational complexity. For each generation, the best N solutions from parent and offspring populations are selected giving the approach an elitist factor. NSGA-II is a generic algorithm and can be implemented for different continuous or binary problems. Jaeggi, Parks, Kipouros, and Clarkson (2006) developed MOTSA for continuous optimization problems. Inspired by path relinking strategies in discrete optimization, the authors proposed an efficient resolution. The objective was to keep the overall MOTSA computational cost at a minimum threshold. Hansen (1997) developed MOTSA using parallel searches. Each Tabu Search algorithm uses variable objective weights and considers a total variable objective equal to a linear weighted sum of the multiple objectives. Each search (thread in practice) performs these weights during the run time with dynamic updates. This strategy is effective if the region of Pareto Front is convex. Otherwise, some Pareto optimal solutions cannot be found by a weighted sum method. Dawson, Jaeggy, Parks, Molina-Cristobal, and Clarkson (2007) developed parallel MOTSA for continuous optimization problems. They compare MOTSA and NSGA-II (developed by Deb et al. (2002)) to test parallel MOTSA efficiency. Considering the authors experiments, parallel MOTSA performs NSGA-II on five test functions out of nine. Jaegyy et al. (2004) adapted MOTSA for real-world optimization problems regarding its handling constraint.

2.3. Multi-objective combinatorial optimization

Concerning multi-objective combinatorial problems, exact methods have very limited performance. Multi-objective simulated algorithm (MOSA) is used in literature to solve different problems. The method is particularly used to solve assignment problems (Tuyttens, Teghem, Fortemps, & Van Nieuwenhuyze, 2000), production scheduling problems (Loukil, Teghem, & Tuyttens, 2005) and packing problems (Ulungu, Teghem, & Fortemps, 1999). Gandibleux and Feville (2000) developed MOTSA for combinatorial problems, and they use dynamic weights updated at every iteration of the neighborhoods' exploration. The algorithm updates each weight proportionally to the deviation of associated objective. After every update, the current list of weight is considered as a tabu. Hansen (2000) developed MOTSA using distance between selected solutions to evaluate the update of every weight at each iteration of the algorithm.

2.4. Multi-objective optimization of MCT handling system

Golias, Theofanis, and Boile (2009) formulated and solved the discrete space and dynamic vessels arrival time. The novelty was to consider the multi-criteria aspect of the problem. Two objectives are maximized: the customer satisfaction and the reliability of the berth schedule. Authors used a multi-objective genetic algorithm to solve the problem. Giallombardo, Moccia, Salani, and Vacca (2010) studied the integrated problem of berth allocation and QC scheduling. Two objectives are considered in which the first is to maximize the total value of chosen QC profiles, and the second is to minimize the housekeeping costs of the transshipment flow. Economical analysis of the value of QC assignment profiles and of yard-related costs in a transshipment context is discussed. Bish et al. (2001) studied for the first time, to the most of our knowledge, the problem of location assignment and AGV scheduling in automated container terminal. The authors proved the NP-Hardness of the integrated problem. The problem was studied as a mono-objective optimization problem with the objective of minimizing the handling time. The vehicle schedule and location assignment are optimized, but the waiting times in bay entry (AGV wait for stacking crane in bay entry) were not considered in this work.

The different proposed approaches analyze only limited parts of the MCT handling system and do not sufficiently cover the set of handling operations in the terminal. Some approaches consider a combination between two chronologically successive optimization problems in MCT, but a limited set of researches considers the multi-objective aspect of these integrated problems. The multi-objective approaches propose at most three-objective optimization models. MOOP are generally treated as a mono-objective problem using a linear function of the different studied objectives. At maritime terminal, MOP are, at the most of the time, non-convex problems. Then if we solve them using a linear function of the considered objectives, some efficient solutions will never be proposed, even if we use a large number of linear objective functions.

MOOP are not sufficiently studied in the general context of MCT and the particular context of container terminal managed by straddle carriers, especially if we consider the number of studied objectives and the approaches of resolution. In our study, we introduce a multi-objective modeling and resolution approach with eight objectives. To solve the problem, we developed a new MOTSA that we will describe later.

3. Traffic layout

In maritime container terminal, at import, when the ship is at the quay, operators have to take next main decisions: For each container in the ship, operators have to allocate a QC to unload it from the ship, a storage location to store it and a straddle carrier to transfer it from the unloading position under QC to the assigned storage location. In this work, we consider the QC scheduling initially known. When straddle carrier accesses a storage bay to finish transferring container to its exact location in that bay, the bay is blocked for some period (superior to the time that straddle carrier need for loading operation at storage location) and no other straddle carrier can access it until the end of that period. When the straddle carrier arrives at the entry of storage bay, if the storage bay is blocked, the vehicle has to respect a waiting time until the end of blocking. Taking into account that blocking constraint, it is important to minimize waiting time at bay entry. The optimization of straddle carriers' routing time and waiting time at bay entry concerns generally the problem of straddle carriers' scheduling. However, the problem of location assignment is directly related to the straddle carrier scheduling problem because when storage locations are assigned to containers, the possible routing paths are limited and depends on the chosen locations. In this work, on the one hand, we integrate the straddle carriers scheduling problem and the location assignment problem which insures higher theoretical optimality, and on the other hand, we study the integrated problem as a multi-objective problem, and we evaluate eight realistic objectives to optimize operating times, storage space organization and the number of required straddle carriers. Optimizing operating times concerns the minimization of all vehicle handling times, the minimization of global operating time called makespan and evaluated as the date of last handling task. Optimization of storage space organization concerns the minimization of the number of required storage bays, the number of assigned storage location, the distance between locations assigned to containers of same client and the number of unproductive moves to be caused by location assignments during future handling operations.

Considering IPLASS, the layout of a container terminal seriously influences the straddle carrier productivity. In fact, the number and the dimensions of storage bays, the number and the length of quays and the number of quay cranes affect the quality of vehicle traffic specifically regarding the routing time.

The traffic layout is another parameter which influences the straddle carrier operating time. This parameter has a second impact which concerns the complexity of the problem. Regarding our approach, the following figure presents the terminal layout considered for our study, where is presented the chosen traffic layout with two common points for all vehicle routing paths: point A at quay entry and point B at quay exit. We present also three possible routing paths depending on assigned storage location. Note that at the quay, straddle carrier traffic has a unique direction from point A to point B. Vehicles are straddle carriers throughout this paper.

In this work, we present a new modeling for IPLASS. We consider general terminal layout, presented in Figure 3, regarding the following properties:

- Only one quay is considered. Then, for multiple quays terminal, we consider the problem only for container ships allocated to the same quay.
- Taking into account QC scheduling, the container-unloading schedule is initially known for each QC. In fact, we have to determine only straddle carrier schedule and storage location assignment. Regarding our layout choices, when storage location assignment is determined, the straddle carrier assignment and schedule are naturally identified (see the end of this part).
- Our modeling supports multiple container-ship unloading operations taking into account compatible arrival times. To support multiple-ship instances, it is crucial to know the exact date of container ship arrivals, the set of QCs used, and the schedule of container handling for each QC.
- The vehicle routing in the quay has to respect a unique direction.

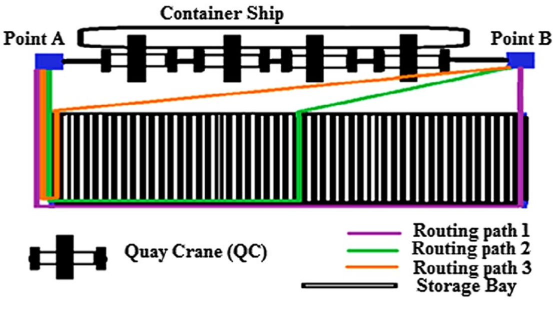


Figure 3. MCT layout for IPLASS.

- Straddle carriers pick up containers under QCs respecting a known global schedule. This schedule is based on the subschedules of containers unloading from ships (QC schedule). If n is the number of QCs used, for every n successive transfer tasks, straddle carriers have to serve all QCs; in other words, QCs are served successively by the fleet of vehicles. QCs operate loading tasks in parallel and do not wait for vehicles arriving. Theoretically, this scheduling constraint influences the problem feasibility and optimality. However, numerical results show that it does not affect the makespan optimality.
- Vehicles are initially located in the preloading position near QCs.
- Every handling task begins when straddle carrier picks up the container under QC.
- Every handling task ends when straddle carrier stacks the container in the associated location.
- Taking into account the straddle carrier traffic, we define the quay entry (Point A in Figure 3) and the quay exit (Point B in Figure 3) as the entry and the output of the part of the quay reserved for QCs used.
- After picking up containers under quay cranes, the straddle carrier moves to quay output; then, it moves to the entry of the bay where the storage location associated with the container is. It waits sufficiently to avoid an accident with the last vehicle entering the same bay, and finally it transfers the container to its storage location in the bay. After the end of this handling task, the vehicle moves to the quay entry and then it moves to the position of the next container picking up task under QC.
- At any time of the process, it is easy to determine which straddle carrier to use for the next container-handling task. In fact, we only have to choose the first vehicle returning to the quay entry (Point A in Figure 3).
- Considering the storage space, the set of free storage locations is initially known; however, we have to determine the storage locations to use for stacking containers. The free storage bays are naturally determined by the free storage locations and the storage bays used are determined by the storage locations used for each solution to the problem.
- With the chosen layout, each handling task is naturally identified by the associated container.

For experiments, we use real databases of “Terminal de Normandie” in the Maritime Port of Le Havre. The terminal is presented in Figure 4.

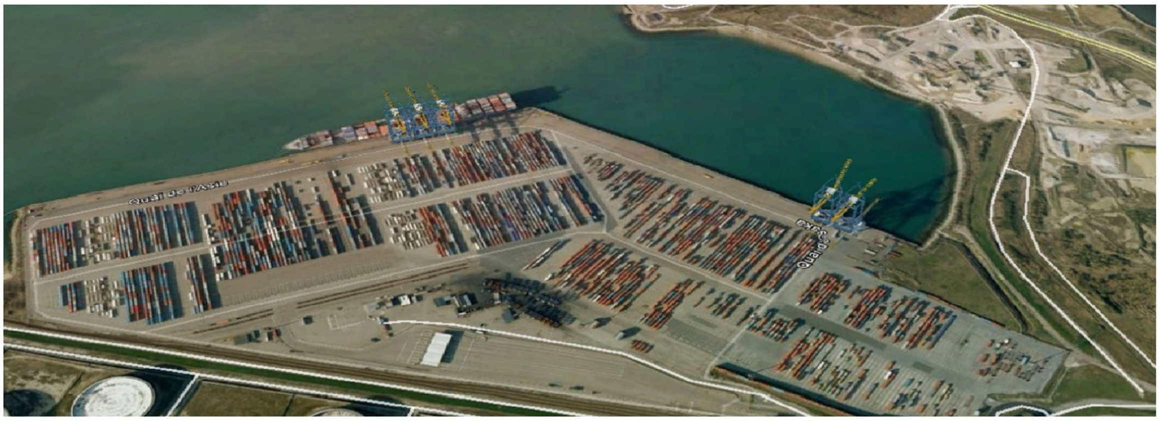


Figure 4 MCT "Terminal de Normandie". Source: Google Maps.

4. Mathematical modeling

Throughout the following, we present mathematical modeling of multi-objective IPLASS in maritime terminal at import. It integrates new and realistic constraints which reflect the real functioning of the terminal. Indeed, we developed constraint formulations to insure an efficient location for the set of containers in order to facilitate some tasks, like the next transportation, the deliveries, or the storage of next arriving containers. These constraints are associated with the evaluation of two location assignment costs.

Constraints associated with the first location assignment cost are essential to facilitate the container transfer to the next transporter (or the delivery) minimizing the total distance among containers of same customer and same delivery date, while those associated with the second one are used to minimize the number of unproductive moves to be caused by location decision.

Our objective is to solve the problem considering the real need of the decider, which is the minimization of the real global operating cost. This cost is mostly considered as a linear function of different components. In some situations, weighted sum methods are efficient to solve MOOP. Generally, if we consider linear objective function, we cannot propose some efficient solutions to the user because the Pareto Front Region is non-convex for combinatorial problem.

Many parameters can influence the operating cost, but the eight chosen objectives represent the most important cost components in IPLASS in a maritime terminal at import.

Operating cost in the general context of MCT concerns essentially storage space resources, equipment resources and operating time resources. Consider now the handling system in container terminal managed by straddle carriers. Storage space resources are bays and locations. Equipment resources are the quay cranes and the straddles carriers. Operating time resources concerns firstly the makespan, which is the date of the last container-handling task (picking up, transfer and storage); secondly, the sum of storage bay occupation times and thirdly the sum of straddle carrier operating times. Taking into account these resources, there is concurrence between the different operations in the terminal and especially for the multi-quay terminals and when operators have to serve many container ships or other transport vehicles at one time. This concurrence influences the weight of the different operating cost components. A global approach can be a good response to this problem of operating cost evaluation. In fact, we can solve the problem for many container ships at one time regarding a global weighted cost.

For the number of free straddle carriers, we can consider that the terminal uses a sufficient number of vehicles to satisfy every container ship. However, the waiting times under QCs make the total number of straddle carriers used in the terminal limited by an upper bound which we evaluate in Section 5.1. Taking into account this limit, operator decision has to take into account the concurrence between container ships for QC resource.

If the terminal has a limited number of straddle carriers, the concurrence between container ships for vehicles is significant for MOO.

The initial storage space configuration is another parameter, which influence the operating cost. It is also an important factor determining the density of feasible solutions in the solution space and the lower bound of the number of storage bay used.

The lower bound of storage bays resource is a very important factor influencing the makespan lower bounds and the resolution hardness. In fact, with a small number of storage bays used, the total straddle carriers waiting time at the bay entry increases considerably.

The number of free storage bays in the terminal at the container ship(s) arrival influences directly the cost of storage space resources and the adequate objective weight.

In our approach, we deal with the operating cost as a vector of eight objective evaluations.

4.1. Data

QC The set of quay cranes used.

$QC(i)$ The quay crane associated with container i. For each container i, $QC(i)$ is initially known.

C The set of tasks (or containers). We can identify each task by its container by considering the known total order of container picking up process.

B The set of free storage bays available for stacking containers.

$B(p)$ The bay of storage location p.

$\mathbf{BE}(p)$ Bay entry of storage location p.

P The set of storage locations. Every location has an initial capacity.

$w(p)$ The initial storage capacity of location p. It is the number of free levels of p. The storage capacity at every location depends on the initial configuration of the storage space. Consider a terminal with a storage space of k **levels,** if the storage location p contains n containers then, for the next handling operation at container ship arrival or departure, the capacity of p is equal to $k - n$**.**

S_{QC} Picking up time under QC. It is the time that the straddle carrier needs to pick up a container under the associated QC. S_{QC} is considered static.

S_v Container storage time. When a straddle carrier arrives at the storage location, S_v is the static time which the straddle carrier needs to stack the container in the associated storage location.

S_B Maximal security waiting time in bay entry. SB is static security parameters. Every straddle carrier has to wait for at most SB seconds in the bay entry. The condition $(S_B > S_v)$ is crucial to eliminate accidents between vehicles entering the same storage bay with an arrival time difference less than S_v.

$\mathrm{succ}_{QC}(i)$ The direct successor of container i considering picking up task under QC.

$T_{p,\mathrm{QC(i)}}$ Straddle carrier routing time from storage location p to QC associated with container i.

$T_{\mathrm{QC}(i),\mathrm{BE}(p)}$ Straddle carrier routing time from QC associated with container i to bay entry of storage location p**.**

T_p Transfer time between the entry of the bay, where is the storage location p, and the exact position of p.

d_i Delivery date of container i to its next transporter (or customer).

d_p Delivery date of the last container stored in location p, regarding the initial storage space configuration.

G A sufficiently big number.

$C(i)$ The set of containers having the same client or the same delivery date as container i.

$T(x, y)$ Routing time between the entry of storage bay x and the entry of storage bay y. This parameter is used to evaluate the first location assignment cost $f_{i,j,x,y}$.

$f_{i,j,x,y}$ The containers allocation cost associated with the decision which stacks container i in storage bay x and container j in storage bay y. $f_{i,j,x,y}$ is initially known data. Note that $f_{i,j,x,y}$ is defined for $i \in C$ and $j \in C(i)$.

$f_{i,j,x,y}$ Equal to $T(x, y)$ if $(x \neq y)$, and equal to S_B if $(x = y)$.

$\tau_{i,j} = 1$ If $d_j > d_i$, 0 else, where i and j are two containers.

$\tau'_{i,p} = 1$ If $d_i > d_p$, 0 else, where i is a container storage location.

4.2. Variables

V The set V represents straddle carriers used. It is an order used to specify every vehicle. $|V|$ is the straddle carrier fleet's size. We consider that $|V|$ can be as large as necessary. $|V|$ is an objective to be minimized in the optimization problem.

V $\{1, 2, \ldots, |V|\}$.

B^* The set of storage bays used for stacking containers. We have to use exactly all these bays. B^* is determined by the storage locations decision. $|B^*|$ is the size of B^*. $|B^*|$ is an objective to be minimized in the optimization problem.

P^* The set of storage location used for storing container after decision. $|P^*|$ is the size of P^*. $|P^*|$ is an objective to be minimized in the optimization problem.

v_i Straddle carrier assigned to container i. $v_i \in V$.

$X'_{i,p}$ Binary variable, equal to 1 if and only if container i is stacked in storage location p.

$X'_{i,p}$ Binary variable, equal to 1 if and only if container i is the first container stored in location p considering the current handling operation.

$V_{i,j}$ Binary variable, equal to 1 if and only if container j is transferred directly after the container i by the same straddle carrier. This variable is defined for $i \neq j$.

$P_{i,j}$ Binary variable, equal to 1 if and only if container j is stored in the same location than container i and directly after i (in others terms, container j is stored directly on container i). $P_{i,j}$ is defined for $i \neq j$.

$P'_{i,j}$ Binary variable, equal to 1 if and only if container j is stored in the same location than container i, directly or indirectly after i (container j is stored on container i, but other containers can be stored between i and j). $P'_{i,j}$ is defined for $i \neq j$.

$B_{i,j}$ Binary variable, equal to 1 if containers i *and* j are stacked in the same bay and container j is stacked directly after i regarding the stacking order in the bay. $B_{i,j}$ is defined only for $i \neq j$.

$t_1(i)$ Start time of task i. The date when the associated straddle carrier picks up container i under QC.

$t_2(i)$ The date when straddle carrier assigned to container i accesses the storage bay of chosen location.

$t_3(i)$ Completion time of task i. The date when associated straddle carrier stores container i in its storage location.

t_v Termination time of vehicle v (straddle carrier v) considering all containers associated with v.

t_b Termination time of container storage in bay b. We consider all containers assigned to storage bay b.

C_{Max} The makespan, which is the date of the last handling task.

I_p If the storage location p is used, I_p is equal to 1, otherwise I_p is nil.

I_b If the storage bay b is used, I_b is equal to 1, otherwise I_b is nil.

$F^7_{i,j}$ A function which evaluates partially the first location assignment cost; equal to zero if containers i and j have neither the same client nor the same delivery date, else equal to the routing time between locations assigned to i and j.

$Z_{i,j}$ Equal to 1 if the location decision assigned to containers i and j will cause an unproductive move, 0 else. These variables are used to evaluate the number of unproductive moves to be caused by location assignment decision, taking into account all containers except those to be stored in first free levels of each location.

$Z'_{\text{i,p}}$ Equal to 1 if i is the first container stored in location p and causes an unproductive move regarding the initial storage space configuration. These variables are used to evaluate the number of unproductive moves to be caused by location assignment decision, regarding only containers to be stored in the first free levels of each location.

4.3. Modeling

The 8 costs of our model are to be minimized under the constraints presented next. All the evaluations of these costs are explained in Section 4.4.

$$\text{Minimize}\left(C_{\text{Max}},\ |C|-\sum_{i\in C}\sum_{\substack{j\in C\\ j\neq i}}V_{i,j},\ \sum_{b\in B}I_b,\ \sum_{p\in P}I_p,\ \sum_{v\in V}t_v,\ \sum_{b\in B}t_b,\ \frac{\sum_{i\in C}\sum_{\substack{j\in C\\ j\neq i}}F_{i,j}/|C(i)|}{|C|},\ \sum_{j\in C}\sum_{\substack{j\in C\\ j\neq i}}Z_{i,j}+\sum_{i\in C}\sum_{p\in P}Z'_{i,p}\right)$$

Location constraints

$$\forall i\in C\colon \sum_{p\in P}X_{i,p}=1 \tag{1}$$

$$\forall p\in P\colon \sum_{i\in C}X_{i,p}\leq w(p) \tag{2}$$

$$\forall (i,p)\in C\times P: X'_{i,p}\leq X_{i,p} \tag{3}$$

Vehicle scheduling constraints

$$\forall i\in C\colon \sum_{\substack{j\in C\\ j\neq i}}V_{ij}\leq 1 \tag{4}$$

$$\forall i\in C\colon \sum_{\substack{j\in C\\ j\neq i}}V_{j,i}\leq 1 \tag{5}$$

$$|C|-\sum_{i\in C}\sum_{\substack{j\in C\\ j\neq i}}V_{i,j}\geq 1 \tag{6}$$

Transfer scheduling constraints in bays

$$\forall i\in C\colon \sum_{\substack{j\in C\\ j\neq i}}B_{i,j}\leq 1 \tag{7}$$

$$\forall i\in C\colon \sum_{\substack{j\in C\\ j\neq i}}B_{j,i}\leq 1 \tag{8}$$

$$\forall (p,\ i)\in P\times C\colon I_p\geq X_{i,p} \tag{9}$$

$$\forall b\in B\colon I_b\leq \sum_{p\in b}I_P \tag{10}$$

$$\forall p\in P\colon I_{B(p)}\geq I_p \tag{11}$$

$$|C|-\sum_{\substack{(i,j)\in C^2\\ i\neq j}}B_{i,j}=\sum_{b\in B}I_b \tag{12}$$

$$\forall b\in B, \forall (i,j)\in C^2, j\neq i\colon B_{i,j}+B_{j,i}\leq \frac{\sum_{p\in b}X_{i,p}+\sum_{f\in b}X_{j,f}}{2} \tag{13}$$

Transfer time constraints

$$\forall i\in C, j=\text{succ}_{\text{QC}}(i)\quad :\quad t_1(j)\geq t_1(i)+S_{\text{QC}} \tag{14}$$

$$\forall (i,p)\in C\times P\colon\ t_2(i)\geq t_1(i)+S_{\text{QC}}+T_{\text{QC}(i),\text{BE}(p)}+G(X_{i,p}-1) \tag{15}$$

$$\forall (i,j)\in C^2,\ i\neq j\colon\quad t_2(j)\geq t_2(i)+S_B+G(B_{i,j}-1) \tag{16}$$

$$\forall (i,\ e)\in C\times P: t_3(i)\geq t_2(i)+T_e+S_v+G(X_{i,e}-1) \tag{17}$$

$$\forall (i,j)\in C^2,\ j\neq i,\quad \forall p\in P: t_1(j)\geq t_3(i)+T_{p,QC(j)}+G(V_{i,j}+X_{i,p}-2) \tag{18}$$

$$\forall i \in C: \quad C_{\text{Max}} \geq t_3(i) \tag{19}$$

Constraints about vehicle attributions and termination time

$$\forall i \in C: \quad 1 \leq v_i \leq |C| - \sum_{i \in C} \sum_{\substack{j \in C \\ j \neq i}} V_{i,j} \tag{20}$$

$$\forall (i,j) \in C^2, j \neq i: \quad v_i - v_j \leq G(1 - V_{i,j}) \tag{21}$$

$$\forall (i,j) \in C^2, j \neq i: |v_i - v_j| \geq 1 - V_{i,j} \tag{22}$$

$$\forall (v,\ i) \in V \times C: \quad t_v \geq t_3(i) - G|v - v_i| \tag{23}$$

Constraint about storage bay termination time

$$\forall b \in B: \quad t_b \geq t_3(i) - G\left(1 - \sum_{p \in b} X_{i,p}\right) \tag{24}$$

Constraint about location costs

$$\forall i \in C, j \in C(i), \forall (b_1, b_2) \in B^2:$$
$$F^7_{ij} \geq f_{i,j,b_1,b_2} - G\left(2 - \sum_{p \in b_1} X_{i,p} - \sum_{l \in b_2} X_{j,l}\right) \tag{25}$$

$$\forall (i,j) \in C^2,\ i \neq j,\ p \in P:$$
$$P_{i,j} + P_{j,i} \leq (X_{i,p} + X_{j,p})/2 \tag{26}$$

$$\forall i \in C: \quad \sum_{\substack{j \in C \\ j \neq i}} P_{i,j} \leq 1 \tag{27}$$

$$\forall i \in C: \quad \sum_{\substack{j \in C \\ j \neq i}} P_{j,i} \leq 1 \tag{28}$$

$$|C| - \sum_{i \in C} \sum_{\substack{j \in C \\ j \neq i}} P_{i,j} = \sum_{p \in P} I_p \tag{29}$$

$$\forall (i,j) \in C^2,\ i \neq j : t_3(j) > t_3(i) + G(P_{ij} - 1) \tag{30}$$

$$\forall (i,j) \in C^2,\ i \neq j: \quad Z_{i,j} \geq 1 + G(\tau_{i,j} + P'_{i,j} - 2) \tag{31}$$

$$\forall (i,p) \in C \times P : X'_{i,p} \geq 1 + G\left(X_{i,p} - 1 - \sum_{\substack{j \in C \\ j \neq i}} P_{j,i}\right) \tag{32}$$

$$\forall (i,p) \in C \times P: \quad Z'_{i,p} \geq 1 + G(\tau'_{i,p} + X'_{i,p} - 2) \tag{33}$$

$$\forall (i,j) \in C^2,\ \ i \neq j:$$
$$P'_{ij} = P_{ij} + \sum_{\substack{k \in C \\ k \neq i \\ k \neq j}} \left\lfloor \frac{P_{i,k} + P_{k,j}}{2} \right\rfloor + \sum_{\substack{f \in C \\ f \neq i}} \sum_{\substack{h \in f \\ h \neq f \\ f \neq j}} \left[\frac{P_{i,f} + P_{f,h} + P_{h,j}}{3} \right] \tag{34}$$

$[X]$: Integer part of real X.

4.4. Objectives

Regarding the case of container terminal managed by straddle carriers, a trial objective is to minimize the makespan which is the date of the last task. But in a real situation, the decision has to take into account another criteria such as the sum of vehicle operating time, the number of straddle carriers used, the number of storage bays used etc. To solve IPLASS in maritime terminal at import, we have two possible approaches. The first is to consider a scalar cost function, which we evaluate taking into account the handling time, the equipment used and the final storage space configuration, and then we proceed for a mono-objective optimization. The second approach is to solve the problem optimizing a vector objective function, taking into consideration all evaluated objectives, and in this case, we consider the strict multi- objective aspect of IPLASS.

We discuss in the following the different objectives to be optimized for IPLASS.

4.5. The number of vehicles to be used—|V|

When we optimize IPLASS, we consider that the terminal has a large resource of straddle carriers, and it can use, for each handling operation at container ship arrival, a sufficiently large number of vehicles to insure makespan high quality or optimality. The only one upper bound which concerns the number of straddle carriers used is the natural parameter of QC productivity (Section 5.1).

We denote the number of straddle carriers to be used by IW and we evaluate it as below:

- $|V| = |C| - \sum_{i \in C} \sum_{\substack{j \in C \\ j \neq i}} V_{i,j}$

Demonstration $\sum_{i \in C} \sum_{\substack{j \in C \\ j \neq i}} V_{i,j}$is equal to the number of containers having direct predecessor (or successor) for a routing task with the same storage straddle carrier. Then $|C| - \sum_{i \in C} \sum_{\substack{j \in C \\ j \neq i}} V_{i,j}$ is equal to the number of containers (or tasks) not having a direct predecessor (or successor). A task with no direct predecessor (or succes- sor) is a first task (or a last task) for some straddle carriers; then, the number of those tasks is equal to the number of vehicles.

If makespan optimality has highest priority, the optimal straddle carrier fleet size is the smallest number which satisfies the next condition:

- For an optimal solution of the problem, when QC finishes unloading a container from a ship, at least one vehicle is ready to pick it up under the QC.

Determining this optimal straddle carrier fleet size is studied by Vis IFA (2000).

4.6. Straddle carriers' makespan—C_{Max}

The straddle carriers' makespan is the date of completion of the last vehicles task. We denote this objective by C_{Max}.

It is a crucial parameter to qualify solutions. The straddle carrier makespan is the global makespan of the handling system. The last straddle carrier's' task is operated when the last container is stored in the storage space.

4.7. The number of storage bays to be used—|B*|

The number of storage bays to be used is an important parameter to qualify the operator decision in MCT using straddle carriers. In fact, at multi-ship arrival, the work in the terminal is organized in different handling operations and the goal of everyone is to transfer containers from a specific part of the quay to the storage space. Considering our modeling, every handling operation concerns one or many container ships but a unique part of the quay. Such an operation is specified by the set of containers to store, the associated part of the quay and its entry and exit points (Point A and Point B in Figure 3—Section 3).

A concurrence between the handling operations concerns especially the storage locations' assignment. When the operator assigns a set of storage locations to containers, a set of storage bays is used. Handling operations cannot use storage bays at the same time without communication. If the frequency of storage bay occupation by straddle carriers is high, the operator cannot use the same storage bays for different handling operations at the same time. Then, it is important to minimize the number of storage bays used for every handling operation. We evaluate this number as below:

- $|B^*| = \sum_{b \in B} I_b$

The number of storage bays used must respect next equation.

- $|B^*| = |C| - \sum_{i \in C} \sum_{j \in C} B_{i,j}.$

Demonstration The same demonstration as in Section 4.4.1.

4.8. The number of storage locations to be used—$|P^*|$

When different handling operations can use the same storage bays, maybe they cannot use the same storage locations in each common bay. The use of a common storage location by different handling operations during a common operating time depends on communication quality and storage strategy. However, the minimization of the number of locations used is an efficient parameter to qualify the operator's decision. We denote this quantity by $|P^*|$ and we evaluate it as below:

- $|P^*| = \sum_{p \in P} I_p$

The number of storage locations used has to respect the next equation.

- $|C| - \sum_{i \in C} \sum_{\substack{j \in C \\ j \neq i}} P_{i,j} = |P^*|$

Demonstration The same demonstration as in Section 4.4.1.

4.9. The total straddle carrier routing time—TSRT

When we minimize the makespan, the operating time is globally optimized. However, if we consider the operating time of every straddle carrier, we have to add another objective which is the sum of the vehicles' operating time. This objective is evaluated as below:

- $\text{TSRT} = \sum_{v \in V} t_v$

4.10. The total bay occupation time—TBOT

When a storage bay is used for a handling operation, the decider has to consider its occupation time. After this time, the storage bay can be easily used for another handling operation. Then, we consider the objective of minimizing the sum of occupation time of storage bays during the current operation. This objective is evaluated as below:

- $\text{TBOT} = \sum_{b \in B} t_b$

4.11. The first location assignment cost—LAC_1

Consider the known data $C(i)$ which is the set of containers having the same delivery date or the same client as container i. We evaluate $F_7(i)$ as an average cost regarding every container j in $C(i)$, $f_{i,j,x,y}$ and $|C(i)|$ the size of $C(i)$.

- $\forall i \in C: F_7(i) = \sum_{\substack{j \in C \\ j \neq i}} \frac{F^7_{i,j}}{|C(i)|}.$

Taking into account $F_7(i)$ for every container i and the number of containers $|C|$, we evaluate the first location assignment cost as below:

- $\mathrm{LAC}_1 = \frac{\sum_{i\in C} F_7(i)}{|C|}$.

For a real instance of the problem, average evaluation of location cost gives a better idea about the quality of the global location decision. The optimization of the first location assignment cost is essential to promote the facilitation of container transfer to the next transporter.

4.12. The second location assignment cost—LAC$_2$

The second location assignment cost is equal to the number of unproductive moves caused by location decision. These unproductive moves will disadvantage the facility of container deliveries. Unproductive moves are caused when a container with some delivery date is stored on other containers with earlier delivery dates; the number of unproductive moves caused by that decision is equal to the number of these containers. This objective is denoted by LAC_2 or UM (Unproductive Moves) and evaluated as below:

- $\mathrm{LAC}_2 = \sum_{i\in C} \sum_{\substack{j\in C \\ j\neq i}} Z_{i,j} + \sum_{i\in C} \sum_{p\in P} Z'_{i,p}$.

4.13. Constraints

Constraints (1), (2) and (3) organize the location decision. For each container, exactly one storage location is associated. The number of containers which are stored in a given storage location p is less than or equal to the capacity of p denoted by w(p). With constraints (4), (5) and (6), the conditions of vehicle scheduling are respected. Each container routing task has at most one successor and one predecessor. The number of vehicles to be used is bigger than or equal to 1. The scheduling conditions in storage bays are ensured by the constraints (7) to (13). In each bay, every container storage task has at least one successor and one predecessor. Constraints (9) ensure the link between location constraints and storage bay ones; it evaluates the number of used locations and then the number of used storage bays. For this evaluation, we add the constraints (10) and (11). Constraints (14) to (19) are injected to bound the different transfer time t_1, t_2 and t_3. Particularly, constraints (19) are added to bound the makespan which is the date of termination of the last handling task. The constraints from (20) to (23) allow the vehicle allocations and evaluate the termination time of every vehicle. The termination of the storage bay operations is bounded in constraints (24). The constraints from (25) to (34) are used to evaluate the location costs F^7_{ij}, Z_{ij} and Z'_{ip}. For more details about the constraints' significance, refer to the in annex 2 of the online supplementary material.

4. 14. NP-completeness of IPLASS

In annex 1 of the online supplementary material, we prove the NP-Completeness of IPLASS. Considering the case of multiple straddle carriers, the demonstration is established using a polynomial time reduction of IPLASS to the Parallel Machines scheduling Problem (PMP). We prove also that the problem is NP-Complete even if we use only one straddle carrier. For that second demonstration, we use a polynomial time reduction of IPLASS to the Traveling Salesman Problem (TSP). More details are given in annex 1 of the online supplementary material.

5. Evaluation of solution quality

Here, in this section, we determine the bounds of our objectives in order to use the obtained lower bounds to evaluate the gaps of the explored solutions. This gap evaluation is introduced to perform the approximation of PF and to qualify the selected solutions.

5.1. Evaluation of |V| upper bound and |V| lower bound

VI is the straddle carrier fleet size. We define the routing cycle as the vehicle routing path including routes from point A to QC, from QC to storage location and from storage location to point A.

The maximal number of vehicles used depends on two factors. The first factor is the maximal productivity of loading tasks under QCs which depends on their speed and their number. The second is the maximal operating cycle, which is the sum of the pick-up time under QC, the maximal routing cycle, and the maximal waiting time in the bay entry. We denote by T_{Max} the maximal operating cycle and by S_{Max} the maximal waiting time in bay entry. S_{Max} depends on the number of QCs and on S_B, where S_B is the maximal waiting time between each straddle carrier and its direct predecessor vehicle operating in the same bay. When a straddle carrier v arrives at the bay entry, the bay can be blocked by one or more vehicles which preceded v in that bay. When v enters the storage bay, the bay is blocked for S_B fixed security time. We suppose now that S_{QC}, the pick-up time under QCs, is bigger than S_B. We denote by $S(v_1, v_2)$ the waiting time in bay entry between a vehicle v_1 and its direct predecessor v_2. We denote by $t_{\mathrm{B}}(v_1)$ and $t_{\mathrm{B}}(v_2)$ the respective arrival times of v_1 and v_2 in bay entry.

- $S(v_1, v_2) = \max\,(0, S_B - (t_B(v_1) - t_B(v_2))$

Then, there is no waiting time in bay entry between vehicles coming from the same QC. In fact, if v_1 and v_2 are served by the same QC, $t_{\mathrm{B}}(v_1) - t_{\mathrm{B}}(v_2) > S_{\mathrm{QC}} > S_B$ then $S(v_1, v_2) = 0$. The result is that the waiting time in bay entry exists only between vehicles coming from different QCs with equivalent routing starting times from QCs.

Consider $t_{\mathrm{QC}}(v_1)$ and $t_{\mathrm{QC}}(v_2)$ the routing starting times from QCs.

- $S(v_1, v_2) = \max\,(0, S_B - (t_{\mathrm{QC}}(v_1) - t_{\mathrm{QC}}(v_2))$

Consider now a vehicle v and all straddle carriers which proceeded v in the same storage bay. If a vehicle v_1 coming from QC_1 affects the total waiting time of v in bay entry, no other vehicle coming from QC_1 can affect it because of pick- up delay under QC_1 which results from the last equation. Then, at most $|\mathrm{QC}| - 1$ predecessor can affect the total waiting time of v in bay entry. We denote it by $S(v)$, and we denote by $\{v_i; i < |QC|\}$ the set of straddle carriers which preceded v in the same storage bay and come from different QCs.

- $S(v) = \sum_{1 \le i < |\mathrm{QC}|} S(v, v_i)$
- $S(v) \le (|\mathrm{QC}| - 1) \times S_B$

We conclude that maximal waiting in bay entry S_{Max} is equal to $(|\mathrm{QC}| - 1) \times S_B$.

- $T_{\mathrm{Max}} = S_{\mathrm{QC}} + \mathrm{Max}\,\{T_{A,p} + T_{p,A}; p \in P\} + S_{\mathrm{Max}}$

Finally, we have an evaluation of the vehicle fleet size upper bound.

- $|V| \le \frac{T_{\mathrm{Max}}}{S_{\mathrm{QC}}} \times |\mathrm{QC}|$

We note that with sufficiently large stacking capacity under QCs, the QCs' productivity is maximal. In other terms, this condition insures the fact that when the QC unloads container from ship, it does not wait for the straddle carriers to unload containers from the stacking space under it.

Suppose now the following two conditions:

- The stacking space under QCs is sufficiently large and QCs productivity is maximal.
- The C_{Max} Optimality is an absolute priority.

Considering these two conditions, we can evaluate a lower bound for the number of straddle carriers used regarding solutions with optimal makespan. Consider S_{opt} a solution having an optimal makespan, the minimal number of vehicles used for S_{opt} depends on three factors: the QCs handling speed which depends on S_{QC}, the number of QCs and the minimal operating cycle T_{Min}. We evaluate T_{Min} as:

- $T_{\mathrm{Min}} = \mathrm{Min}\,\{T_{A,p} + T_{p,A}; p \in P\} + S_{\mathrm{QC}}$

Moreover, we have next the inequality:

- $|V(S_{\mathrm{opt}})| \ge \frac{T_{\mathrm{Min}}}{S_{\mathrm{QC}}} \times |\mathrm{QC}|$

Consider now the last two conditions added to the next proprieties.

- We suppose the existence of C_{Max}-Optimal solution S_{opt} with nil waiting times in bay entries.
- There is no possibility of short cut paths between the quay entry (point A) or the quay exit (point B) and the storage bay. All storage bays have the same length. In other terms, all straddle carrier routing paths are equal.

With these conditions, the inequality becomes an equality and we obtain a strict evaluation of $|V(S_{\mathrm{opt}})|$-Optimality.

- $|V(S_{\mathrm{opt}})| = \frac{T_{\mathrm{Min}}}{S_{\mathrm{QC}}} \times |\mathrm{QC}|$

5.2. Evaluation of C_{Max} lower bound

In this part, we denote by Makespan (S) the value of C_{Max} taking into account solution S. It is the date of the last task for the decision S. Consider V^* the minimal set of vehicles sufficient to insure makespan optimality

In our modeling, for each vehicle, the routing path includes the routing path from the QC unloading the container to the storage location, and the routing path from the storage location to the QC associated with the next container to transfer. In this part, we consider that routing path include the routing path from point A (quay entry) to the storage location and the routing path from the storage location to point A. This consideration does not affect the nature of the problem and its optimality. Consider p_0 the storage location with the shortest routing cycle. We denote by T_0 the routing time of the shortest routing cycle added to loading time at QC and unloading and stacking time at storage location. We denote by N_0 the maximal number of containers which the largest fleet of straddle carriers can transfer in the routing time T_{Min}. In mathematical terms, we have:

- $T_{\mathrm{Min}} = \mathrm{Min}\,\{T_{A,p} + T_{p,A}; p \in P\} + S_{\mathrm{QC}} + S_B$
- $N_0 = |\mathrm{QC}|\frac{T_0}{S_{\mathrm{QC}}}$

With $|\mathrm{QC}|$ the cardinal of QCs set, A the point at quay entry and $T_{A,p}$ The routing time between A and storage location p and $T_{p,A}$ the routing time between p and A.

- We consider the storage location p associated with the minimal total routing path $\mathrm{Min}\,\{T_{A,k} + T_{k,A}; k \in P\}$.
- We consider $w(v, i)$ the vehicle waiting time in bay entry. It is exactly the waiting time for vehicle v when it transfer container i.
- V^* is the minimal set of vehicles which insure maximal productivity for the speed of containers picking up under QCs. V^* is evaluated considering that waiting times in bay entry are nil. Then we can conclude that $|V^*| = N_0$.
- Makespan $(S) \ge \frac{|C|}{|V^*|} \times T_{\mathrm{Min}}$

$$\mathrm{Makespan}\,(S) = \mathrm{Max}\{\sum_{i \in C(v), j = \mathrm{succ}_v(i)} (T_{\mathrm{QC}(i),p(i)} + T_{p(i),\mathrm{QC}(j)}$$
$$+ S_{\mathrm{QC}} + S_V + w(v, i)); v \in V^*\}$$

$$\geq \text{Max}\left\{ \sum_{i\in C(v)} (T_{Ap(i)} + T_{p(i),A} + S_{\text{QC}} + S_v); v \in V^* \right\}$$
$$\geq \text{Max}\left\{ \sum_{i\in C(v)} \text{Min}\left\{ T_{A,p(i)} + T_{p(i),A} + S_{\text{QC}} + S_v; i \in C(v) \right\}; v \in V^* \right\}$$
$$\geq \text{Max}\left\{ |C(v)|\text{Min}\{ T_{A,p} + T_{A_{J,p}} + S_{\text{QC}} + S_v; p \in P\}; v \in V^* \right\}$$
$$\geq \text{Max}\left\{ |C(v)|; v \in V^* \} \times \text{Min}\{ T_{A,p} + T_{p,A} + S_{\text{QC}} + S_v; p \in P \right\}$$
$$\geq \frac{|C|}{|V^*|} T_{\text{Min}}$$

Suppose that the stacking space under QCs has sufficiently large capacity to insure QCs' operating time optimality. With Journal of the Operational Research Society that condition, we can evaluate the date of the last QC handling task with the next formulation. We denote this quantity by Makespan(QC$_{\text{opt}}$) and we denote by C(q) the set of containers associated with QC q.

- Makespan $(\text{QC}_{\text{opt}}) = \text{Max}\ \{|C(q)| \times S_{\text{QC}}, q \in \text{QC}\}$

Regarding the same conditions, we evaluate another lower bound for the makespan of the global handling system. Suppose that all routing paths correspond to the same routing time T. S is a feasible solution to the problem.

- Makespan $(S) \geq$ Makespan $(\text{QC}_{\text{opt}}) + T$

In our modeling, we consider a regular QC unloading task with a static container unloading time less than the straddle carrier picking up time S_{QC}. A total container picking up schedule is initially considered. For the most general context, we evaluate the next inequality for each solution S and for each QC handling situation.

- Makespan $(S) \geq \frac{|C|}{|\text{QC}|} \times S_{\text{QC}} + (|C| \mod |\text{QC}|) \times S_{\text{QC}}$

Consider now the set of storage bays used B^*. We denote by b the storage bay in B^* containing the largest set of storage locations used. We suppose the productivity of straddle carriers maximal. We denote by T the routing time associated with each container transfer to b. In these conditions, we have the following equation:

- Makespan $(S) = (|C(b)| - 1) \times S_B + T$, where $|C(b)|$ is the number of containers to store in b.

Considering decision S, B^* and b are initially unknown. At least b contains $|C|/|B^*| + |C| \mod |B^*|$ containers. Suppose that all routing paths correspond to the same routing time T. Then, for each solution S, we have the following inequality.

- Makespan $(S) \geq (\frac{|C|}{|B^*|} - 1) \times S_B + |C| \mod |B^*| + T$

5.3. Evaluation of $|B^*|$ lower bound

The lower bound, taking into account the number of bays used, is equal to the cardinality of the smallest set of bays which contains a set of free locations with a total storage capacity equal to the number of containers to be stacked.

In mathematical terms, we have next evaluation of $|B^*|$ Lower Bound:

- $LB(|B^*|) = \text{Min}\left\{ |F|; F \subseteq B \text{ and } \sum_{b\in F}\sum_{p\in b} w(p) = |C| \right\}$

5.4. Evaluation of $|P^*|$ lower bound

The lower bound, regarding the number of used storage locations, is equal to the cardinal of the smallest set of locations having a total storage capacity equal to the number of containers to be stored. Mathematically, we have next valuation of $|P^*|$ Lower Bound.

$$\text{LB}\,(|P^*|) = \{|H|; H \subseteq P \text{ and } \sum_{p\in H} w(p) = |C|\}$$

5.5. Evaluation of TSRT lower bound

Consider P^*_{Min} the set of storage locations used. We suppose that P^*_{Min} contains the $|C|$ storage locations which corresponds to the $|C|$ shortest routing paths. For each solution S, we have the following inequality.

- $\sum_{v\in V} t_v \geq \sum_{p\in P^*_{\text{Min}}} (T_{A,p} + T_{p,A} + S_{\text{QC}} + S_v)$

Suppose that every routing path have the same length T; then, we obtain the next result.

- $\sum_{v\in V} t_v \geq |C| \times (T + S_{\text{QC}} + S_v)$

5.6. Evaluation of TBOT lower bound

Let T_b denote the routing time between the entry of the bay b and the quay entry (Point A). Consider the storage bay b in B^*, we know that $t_b \geq (|C(b)| - 1) \times S_B + T_b$ (Section); then, we can conclude next inequality.

- $\sum_{b\in B^*} t_b \geq (|C| - |B|^*) \times S_B + \sum_{b\in B^*_{\text{Min}}} T_b$

Suppose that all routing paths have the same length which corresponds to the operating time T. We obtain the next result.

- $\sum_{b\in B^*} t_b \geq (|C| - |B^*|) \times S_B + |B^*| \times T$

5.7. Evaluation of lower bound for the first location assignment cost

The minimal distance between two different storage locations is a lower bound of the first location assignment cost.

5.8. Evaluation of lower bound of the second location assignment cost

Regarding realistic initial storage space configuration, the lower bound of the second location assignment cost is equal to zero.

6. Multi-objective optimization and Pareto optimality

For many optimization problems, to take a decision, we have to satisfy different criteria. These problems are called multi-criteria optimization problems or multi-objective optimization problems (MOOP). Taking into account the different criteria of a MOOP, a solution x can be better than another solution y according to some criteria and worse according to others. Generally, there is no unique optimal solution for a MOOP, but a set of efficient solutions called the Pareto Front. These solutions are qualified as Pareto optimal or efficient.

A MOOP can be represented in the decision space or the criterion space. In criterion space, MOOP can be mathematically represented as:

$$\min q \quad q \in Q$$

where Q represents the set of feasible points in criterion space (feasible region) and q is the vector of n criterions function called also objective function. In the case of IPLASS, Q is defi taking into account the evaluation of each objective to be optimized. In fact, we deal with a multi-criteria evaluation problem.

Consider now the decision space, S the set of feasible solutions, n the number of objectives (or criteria) to be optimized, f_i $(1 \le i \le n)$ a scalar function. MOO can be represented mathematically as:

$$\min \left(f_1(x), f_2(x), \ \ldots, f_{n(x)}\right) \quad x \in S$$

When we solve a multi-objective problem (or a multi-criteria problem), we cannot consider directly the ordinary scalar optimality. In fact, a Pareto optimality concept is defined.

- $\forall (x, y) \in S^2, x \neq y$: x weakly Pareto dominates y if and only if: $\forall i \in \{1, 2, ..., n\} : f_i(x) \le f_i(y)$ and $\exists i \in \{1, 2, ..., n\}$:$f_i(x) < f_i(y)$
- $\forall (x, y) \in S^2, x \neq y$:$x$ strongly Pareto dominates y if and only if: $\forall i \in \{1, 2, ..., n\} : f_i(x) < f_i(y)$
- A solution x is called Pareto optimal or efficient, if and only if their does not exist another solution y which weakly Pareto dominates x.
- A solution x is weakly Pareto optimal or efficient if and only if their does not exist another solution y which strongly Pareto dominates x.

7. Multi-objective Tabu Search algorithm—MOTSA

To solve IPLASS, we developed a new MOTSA. Our resolution is a cyclic opportunist exploration of the solution space considering a specific neighborhood. We defined the neighborhood taking into account the different goals to be optimized. The exploration of solution space is composed of different cycles, and every cycle is composed of different periods. The exploration strategy is based on two main concepts: a periodic moderation of objective weights used to evaluate a variable objective function, and a selection of distant solution at the beginning of each period.

The initial solution is elected from a sufficiently big set of feasible solutions. Regarding a current solution, the best neighbor is determined using weighted sum method. Each objective weight is a cross of two variables, the first is a cost updated at the beginning of every cycle and the second is a sufficiently small value updated at the beginning of every

period. Considering that the Pareto Front Region is non-convex; at the beginning of each period, a distant element is selected. At every neighborhood exploration, the elected neighbor is added to a tabu list and declared tabu during the current period. At the end of every period, the tabu list is cleared and initialized with a single element which is the distant solution.

Because the problem is a mixed-integer problem, we do not consider only the best neighbor when we update Pareto list, but also all potentially efficient solutions in every neighborhood to approach more efficiently the non-convex Pareto Front Region. Every potentially efficient solution in the current neighborhood is added to the Pareto list. After every solution injection, the Pareto list is updated and each dominated solution is deleted from the list.

Termination conditions are described in Section Four indicators of efficiency and 2*D*-projection of the approximated Pareto Front are evaluated in order to facilitate the user decision.

Note that our MOTSA gives an approximation of the PF, and then each given solution is not necessarily Pareto optimal but potentially Pareto optimal. Note also that during solution space exploration, we declare a current solution as potentially efficient if and only if that solution is not dominated by another solution explored during the previous iterations.

7.1. Solution representation

For the most general context of IPLASS, each solution is a four-dimensional vector and every variable of this vector comprises four integers which represent a task, a container, *a* vehicle and a storage location. With our traffic layout particularities (at every time the vehicle to use for the next container transfer is the first straddle carrier returning to quay entry) and established container transfer schedule, the decisional problem concerns only the choice of storage places for every container. And with that layout, we can establish a total container order without changing optimality for general real configuration of storage space. We conclude that we can use a data list to represent solutions. Consider an instance of 10 containers and 30 storage places. The solution *S*: ((1, 10), (2, 13), (3, 3), (4, 1), (5, 0), (6, 6), (7, 7), (8, 22), (9, 29, (10, 5)) can be presented by the list (10, 13, 3, 1, 0, 6, 7, 22, 29, 5).

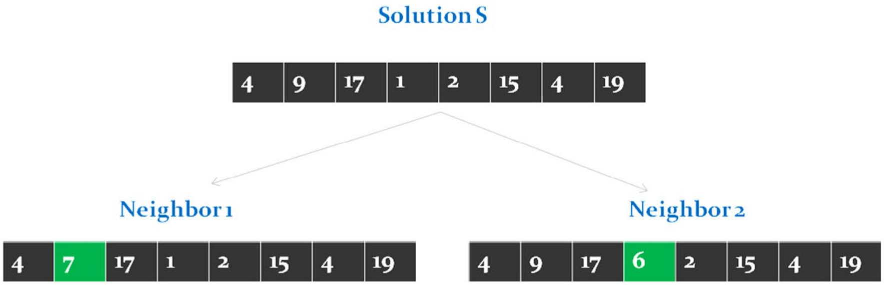


Figure 5. Neighborhood considering storage location aspect.

7.2. Neighborhood construction

The neighborhood is composed of three different sub-neighborhoods: neighborhood considering storage location aspect, neighborhood considering storage bay aspect and neighborhood considering the number of straddle carriers.

7.3. Neighborhood considering storage location aspect

For every solution, the global neighborhood contains $|C|^*(|P| - 1)$ elements. Taking into account the real dimensions of instances, it is not effective to select all the solutions during the exploration of each neighborhood. Only a sufficient random part of the neighborhood is considered. Consider a solution $S = (P_1, P_2, \dots, P_{|C|})$, where P_i is an element of P for each integer i between 1 and $|C|$. Consider $V(S)$ the neighborhood of S, then:

$$\forall N \in V(S),\ N = (N_1, \dots, N_{|C|}):\ \exists! i \le |C|,\quad N_i \ne P_i$$

In Figure 5 we present two possible neighbors for solution S considering storage location aspect.

7.4. Neighborhood considering storage bay aspect

Consider now a solution S, $B(S)$ the set of bays used regarding solution S and B(Si) the bay of storage location assigned to container number i.

$$N \in V^+$$

$$\Leftrightarrow$$

$$B(S) \subset B(N)$$

$$|B(N)| = |B(S)| + 1$$

$$\forall i \le |C| \text{ if } B(N_i) \in B(S) \text{ then } N_i = S_i \text{ else } N_i \ne S_i$$

$$N \in V^-$$

$$\Leftrightarrow$$

$$B(N) \subset B(S)$$

$$|B(N)| = |B(S)| - 1$$

$$\forall i \le |C| \text{ if } B(S_i) \in B(N) \text{ then } N_i = S_i \text{ else } N_i \ne S_i$$

In Figure 6, we present two possible neighbors of a giving solution S.

7.5. Neighborhood considering straddle carriers used

Taking into account solution S using n vehicles, two neighbors are possible: one with $n + 1$ (if n is smaller than maximal number of straddle carriers used) vehicles and another with $n - 1$ vehicles (if n is larger than 1).

7.6. Initial solution election

The initial solution is elected considering a set of n solutions. The size of this set depends on instance size and especially on the number of containers to store.

7.7. Cycles and periods

The exploration of solution space is composed of different cycles where every cycle is composed of different periods. Each period is composed of a set of neighborhood explorations.

7.8. Objective weights and distant solutions

The developed MOTSA is based on a cyclic opportunist exploration. The updated objective weights determine the inlluence of every objective at every exploration step. For every objective k, the weight W_k is evaluated as follows: $W_k = \alpha_k \times C_k$

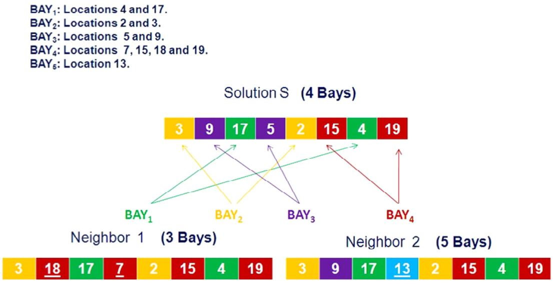


Figure 6. Neighborhood considering storage bay aspect.

α_k The α-weight of objective k. The α-weights are updated at every new period. To update α-weights, we use weighted sum method explained in Section 7.7.

C_k The cost of objective k during the current cycle. C_k has to be sufficiently large compared to the α-weights. We use objective cost C_k in order to perform the classic weighted sum method. In fact, we regulate the exploration of approximated Pareto Front with strategy of compromise between objectives and variation of the influence of each objective unity (the influence oflosing or earning one unity of handling times or vehicles used or other objective).

At the beginning of every period, we select a neighbor which represents a maximal distance regarding the Pareto list. We can describe this neighbor as a distant element. We use next function for that selection.

From the neighborhood of solution S, we select the solution which maximizes next distance d evaluated as: $\forall v \in V,\ d(v) = \sum_{k \in IK} \min \left\{ \frac{|F_k(v) - F_k(w)|}{h_k},\ w\ \in\ L \right\}$, where L is the Pareto list, IK is the set of objectives and h_k is the difference between the maximal and the minimal values of objective k considering the current solutions of L.

7.9. Objective costs update

Every cycle is composed of a set of periods. At the beginning of each cycle, the cost of every objective is updated. To update the different costs, we consider two factors; the first is the deviation of the objectives from their lower bounds taking into account the solutions in the current Pareto list; the second is the values of objective costs during the precedent cycles. We describe the cost of each objective k as an objective cost and we denote it by c_k.

For the first factor influencing the update of objective cost c_k, we consider all the elements of Pareto list and we define the average gap of objective k $\widetilde{\text{gap}}_k$ as follows:

$$\widetilde{\text{gap}}_k = \frac{\sum_{S \in \text{PL}} \text{gap}_k(S)}{|\text{PL}|}$$

where $|\text{PL}|$ is the number of solutions in Pareto list PL and $\text{gap}_k(S)$ the gap of solution S regarding the objective k. For k between 1 and 7, we evaluate this gap as follows:

$$\text{gap}_k\,(S) = \frac{F_k(S) - \text{LB}_k}{\text{LB}_k}$$

$F_k(S)$ The value of objective k taking into account solution S

$\mathbf{LB}_k$ The lower bound of objective k

For $k = 8$, we evaluate the gap of eighth objective as follows: $\text{gap}_8(S) = F_8(S)$.

After the evaluation of the average gaps of Pareto list PL, we store them in a list L^G in ascending order and we define the first priority of objective k as follows:

$$\text{priority}_k^1 = \text{ index of } \widetilde{\text{gap}}_k \text{ in } L^G$$

For the second factor influencing the update of objective cost c_k, we consider all precedent values of c_k for each objective k. We evaluate the average cost of objective k (denoted by k) as follows:

$$\tilde{c}_k = \frac{\sum_{u \in U} c_k^u}{|U|}$$

U The set of precedent cycles.

c_k^u The cost of objective k during the cycle u.

After the evaluation of the average cost of every objective k (AC_k) during the precedent cycles, we store them in a list L^U in descending order and we define the second priority of objective k (priority_k^2) as follows:

$$\text{priority}_k^2 = \text{ index of } \text{AC}_k \text{ in } L^U$$

The new cost of each objective k is evaluated as follows:

$$c_k = 10^{2 \times \text{priority}_k^1 + \text{priority}_k^2}$$

7.10. α-Weights update

At the beginning of each cycle, the cost of every objective is updated; then, these costs are fixed and no other update is possible until the end of the cycle. However, during the cycle, weighted sum method is applied to more diversify the exploration of Pareto Front.

During each cycle, at the beginning of every period, the α-weights are updated. Update process is based on two considerations; the α-weights history during current cycle and the deviation of the objectives from their lower bound during previous periods in current cycle.

For each objective, the new updated α-weight is a perturbation of its last α-weight. Perturbation can be positive or negative with a probability which depends on the average gap ($\widetilde{\text{gap}}$) and the average α-weight ($\tilde{\alpha}$) of the objective. Average gap is evaluated as in Section 7.6 Average α-weight is evaluated for each objective k as follows:

$$\tilde{\alpha}_k^{c,p} = \frac{\sum_{m \in \text{pred}_p^c} \alpha_k^{c,m}}{|\text{pred}_p^c|}$$

c current cycle

p Current period in cycle c

pred_p^c The set of periods, preceding period p during cycle c

$|\text{pred}_p^c|$ The size of pred_p^c

$\tilde{\alpha}_k^{c,p}$ The average weight of objective k considering pred_p^c

$\alpha_k^{c,m}$ The α-weight of objective k during period m of cycle c

After the evaluation of the average weight of every objective k during the periods preceding period p in current cycle c, we store them in a list L^W in descending order and we defme the third priority of objective k (priority_k^3) as follows:

$$\text{priority}_k^3 = \text{index of } \tilde{\alpha}_k^{c,p} \text{ in } L^W$$

At the beginning of period p, the probability of positive deviation of the weight associated with objective k, Prb_k^p, is evaluated as next:

$$\text{Prb}_k^p = \frac{2 \times \text{priority}_k^1 + \text{priority}_k^3}{3 \times |IK|}$$

|IK| the number of objectives to be optimized.

For the objective having favorable Average Gap (AG) compared to the other objectives, the probability of negative perturbation of the associated weight is greater especially if the Average value of the associated weight is significant. On the contrary, the probability of positive deviation is greater.

After each update, the vector of new α-weights (α = (α_1,…, $\alpha_{|IK|}$)) is added to the tabu list of weight used. When the tabu list size limit is reached, as for classical tabu list, the oldest vector of weight is removed.

At the end of every cycle, the tabu list of α-weight vectors is initialized (all weight vectors are removed from the tabu list).

We present next the algorithm of the described process of weight update.

Δ The absolute value of the sum of negative perturbations.

Q The number of positive perturbations

p The current period

σ_k^p The sign of the perturbation of objective k during period p

α^p The vector of objective weighs in period p. $\alpha^p = (\alpha_1^p, \alpha_2^p, \ldots, \alpha_k^p, \ldots, \alpha_N^p)$

L_W The Tabu list of weight vectors

N The number of objectives

Solution proposed by indicator 1 🟥

Solution proposed by indicator 2 🟩

Solution proposed by indicator 3 🟦

Solution proposed by indicator 4 ⬛

Note that "oldestvector (L_W) " is a function giving the oldest element injected to the list of weights L_W.

7.11. Pareto list update

After every election of a new current solution S, if S is not dominated by Pareto list elements, then it is added to the Pareto list, and Pareto solutions dominated by it are removed from the list. Taking into account the non-convexity of Pareto Front Region, we apply the same process to the elements of partial neighborhood of S.

7.12. Finalization conditions

The developed MOTSA uses three kinds of iterations: cycle iterations, period iterations and exploration iterations. A negative iteration is an iteration which does not affect the Pareto list, while positive iteration is an iteration which affects the Pareto list.

The end of each period is when the number of successive negative explorations, N, is equal to its maximal tolerable value N_{Max}, or when the total number of neighborhood explorations, E, is equal to its maximal tolerable value E_{Max}.

The end of each cycle is when the number of successive negative periods, M, is equal to its maximal tolerable value M_{Max}, or when the total number of neighborhood explorations, E, is equal to its maximal tolerable value. The end of MOTSA resolution process is when the number of successive negative cycles, Q, is equal to Q_{Max} or the number of total neighbor- hood explorations is equal to E_{Max}.

For experiments, we change the second condition ($E < E_{\text{Max}}$) with a maximal run time. The user can also stop the process if he is satisfied by the set of solutions proposed by indicators of efficiency (Section 7.11).

7.13. MOTSA

7.13.1. Variables and constants

IK The set of objectives.

|IK| The cardinal of IK, it is the number of objectives to optimize.

C_k The cost of objective k during current cycle.

α_k The α-cost of objective k during current period.

E The total number of neighborhood explorations.

N The number of successive negative explorations considering last neighborhood explorations.

M The number of successive negative periods regarding last periods.

Q The number of successive negative cycles taking into account last cycles.

M_{Max} , N_{Max} , E_{Max} , Q_{Max} Maximal tolerable values of M, N, E. and Q.

Negative cycle, negative period, negative exploration Boolean variables used to detect respectively the end of MOTSA, the end of cycle and the end of period, considering the respective values of Q, M and N.

S Current solution. At every iteration of the exploration process, the current solution is the element elected from the last explored neighborhood.

elite neighbor (S) Function generally returning an elected neighbor which is the best neighbor regarding the current solution S and the current objective coefficient regulation. The elected element is the neighbor which minimizes the weighted sum function as follows:

$$\text{Minimize} \sum_{k \in IK} C_k \alpha_k F_k(v),\ \ v \in V_h(S)$$

If that element is non-tabu it is returned as a result, else if it satisfies the aspiration condition ($h(N) \geq h_p$) it is returned with a probability of 0.5. If the function does not return the best neighbor, it returns the second best neighbor or the third with the same conditions.

L Pareto list which contains the current set of solutions approximating the Pareto Frontier. MOTSA updates this list at every neighborhood exploration (positive exploration).

T The tabu list.

$S.h$ Indicator of exploration efficiency considering solution S and current Pareto list. h is equal to the number of solutions delated from the Pareto list when the solution S **is** injected into it. $S.h$ is also denoted by $h(S)$. Note that $S.h$ is used as an indicator of efficiency during the algorithm execution, and after the end of solution space exploration, other indicators of efficiency are used to select specific solutions.

h_p A limit value to identify promising solutions. This value is used to establish the aspiration strategy.

$V(S)$ Partial neighborhood of S. The optimal size of $V(S)$ depends on the instance size. To determine that size, we executed several simulation tests.

$V_h(S)$ A partial neighborhood of S with a size depending on $h(S)$. **If** $H(S) \geq H_p$, then $|V_h(S)| = 2\,|V(S)|$ else $V_h(S) = V(S)$.

7.13.2. Algorithm

```
Q ← 0, E ← 0, S ← initial solution , L ← {S}
for k ∈ IK: $C_k$ ← 1
for k ∈ IK: $\alpha_k \leftarrow \frac{1}{|IK|}$
While (Q ≤ $Q_{max}$) and (E ≤ $E_{max}$)
{ M ← 0,  negative cycle ← true
 if (E>0) {update objective costs, for k ∈ IK: $\alpha_k \leftarrow \frac{1}{|IK|}$}
  While (M ≤ $M_{max}$) and (E ≤ $E_{max}$)
  {N ← 0, negative period ← true
   if (E > 0 ) { if (it is not first period of the cycle) {update α-weights}, S ← distant Element (V(S), L), T ← {S} }
    While (N ≤ $N_{max}$) and (E ≤ $E_{max}$)
    { E ← E + 1, negative exploration ← true, H ← 0
      if (S is not dominated by any solution in L)
      {         for x ∈ L
                if (x is dominated by S)     {   L ← L\{x}, S.h ← S.h + 1   }
                if (S is not in L)   L ← L ∪ {S}                                  }
                if (S is not in tabu list)     T ← T ∪ {S}
                          for v ∈ $V_h$(S)
                          if (v is not dominated by solutions of L)
                          {         if (negative exploration = true)
                                    negative exploration ← false
                                    if (negative period = true )
                                    negative period ← false
                                    if (negative cycle = true )
                                    negative cycle ← false
                                    for x ∈ L
                                    if (x is dominated by v)
                                    {    L ← L\{x}, v.h ← v.h + 1
                                         L ← L ∪ {v}                    }
      if (negative exploration = true)       N ← N + 1 else N ← 0
      S ← elite neighbor(S)                                            }   END WHILE
    if (negative period = true) M ← M + 1 else M ← 0                        }        END WHILE
  if (negative cycle = true) Q ← Q + 1 else Q ← 0                                     }        END WHILE
```

Lines (1), (2) and (3) Initializations of Q, E, current solution S, Pareto list L, objective costs and α-weights.

Lines (4), (7) and (10) Respectively the fization conditions of MOTSA, cycles and periods.

Line (5), (8) and (11) Respectively the initializations at the beginning of each cycle, period and neighborhood exploration.

Line (6) Objectivecostupdateandα-weight initialization at the beginning of every cycle except the first cycle.

Line (9) α-Weight update and selection of distant element at the beginning of each period except the first period (Sections 7.5 and 7.7).

Line (13), (14) Update of Pareto list L if the condition of and (15) line (12) is true.

Line (16) Update of tabu list T.

From (17) to (28) For every neighbor v in $V_h(S)$, if v is not dominated by the solutions in Pareto list L, the current exploration, period and cycle are declared positives, the indicator $v.h$ is evaluated and L is updated.

Lines (25), (26) and (27) Pareto list update after selection of solution non-dominated by the elements of the list.

Line (28) Injection of selected solution in the Pareto list L.

Lines (29), (31) and (32) If the current step (the cycle, the period or the exploration) is declared negatively, the associated counter is incremented else it is initialized to zero.

Line (30) S is updated for next exploration.

7.14. Solution election

Considering the Pareto list elements, we elect a set of solutions using four efficiency indicators (EI). The first indicator of efficiency, denoted by EI_1, is evaluated as follows:

$$\mathrm{EI}_1(S) = \sum_{k\in\mathrm{IK}} \sum_{p\in\mathrm{PL}} I_{S,p,k}$$

where $I_{S,p,k}$ equal to 1 if $F_k(S) \le F_k(p) - \delta_k$, equal to 0 else. IK is the set of objectives, PL the Pareto list and $F_k(S)$. the value of objective k taking into account objective function F and solution S. δ_k is a constant depending on the smaller significant variation of objective k. For experiments, the value of δ_k depends on the objective. For the first, fifth and sixth objectives (objective evaluating operating times): $\delta_1 = \delta_5 = \delta_6 = 6$ s. For the second, the third, the fourth, the seventh and the eighth objectives (objective minimizing the number of equipment used and the location assignment cost): $\delta_2 = \delta_3 = \delta_4 = \delta_8 = \delta_7 = 0$ unity.

The second indicator of solution efficiency, denoted by EI_2, is equal to the sum of the orders of preference of the solution regarding the different objectives. Consider solution S with the respective orders of preference I_1 until I_8 taking into account the 8 objectives to be minimized, the second indicator of efficiency is equal to the sum of these orders:

$$\mathrm{EI}_2 = \sum_{1\le k\le 8} I_k(S).$$

The best solution regarding the second indicator of efficiency is the one which minimizes it.

The third efficiency indicator, denoted by EI_3, is a linear function with choosing objective weights. The best solution for the third indicator is the one which minimizes that chosen function. To choose efficient weights, we must consider the difference between the objectives in their intervals of variations. For example, we cannot accept a big number of unproductive moves to win only some seconds of operating time. In fact, the chosen weights must establish some equilibrium between the different objectives.

The fourth indicator of efficiency, denoted by EI_4, is a function of the three first indicators. Consider a solution S and O_1, O_2 and O_3 the respective orders of efficiency of S taking into account all solutions in approximated Pareto Front for the first, the second and the third indicators. The value of the fourth indicator is equal to the sum of these orders of preference.

$$\mathrm{EI}_4\,(S) = \mathrm{O}_1 + \mathrm{O}_2 + \mathrm{O}_3.$$

The best solution regarding the fourth indicator is the one that minimizes it.

The first and the second indicators of efficiency are theoretically adapted to the combinatorial aspect of the problem as a non-convex problem with a non-convex Pareto Front Region. The third indicator of efficiency is easy to be adapted by operators in the container terminal taking into account the current situation and especially the current priorities. The fourth indicator of efficiency is able to give a solution taking into account the combinatorial aspect of the problem and the preferences of the operator.

To elect a solution from the Pareto list, we used 2D-projections of the multi-objective space and we select one of the solutions proposed by the different indicators of efficiency.

Table 2. Lower bound evaluations.

Objective	C_{Max}	$\|V\|$	$\|B^*\|$	$\|P^*\|$	$TSRT = \sum_{v\in V} t_v$	$TBOT = \sum_{b\in B} t_b$	LAC_1	LAC_2
Lower bound	27110 s	18	11	112	465465 s	14000 s	1s	0

8. Numerical experiments

In this part, objectives are denoted as in Section 4.4

8.1. Approximated Pareto Front for large instance

Table 4, in next part, shows four elected solutions given by MOTSA for a large instance of 1000 containers and a stacking space with a total storage capacity supporting 10000 containers. More details are given in annex 3 and annex 4 of the online supplementary material with all solutions proposed to approximate Pareto Front (more than 5000 solutions). The next computational results are obtained after 3000 neighborhood explorations with 805626 explored solutions. Lower bounds are determined to evaluate resolution quality. For this instance, we evaluated lower bounds presented in Table 2.

Remarks: We modified the maximal capacity of storage locations (these of Terminal de Normandie in order to treat large instances of the problem. For solution space exploration, we refuse any solution with makespan gap upper to 0.1. That is an additional constraint considered to respect a maximal deviation of 2700 *s* from the makespan lower bound.

Note that lower bound of IW is evaluated considering an optimal makespan.

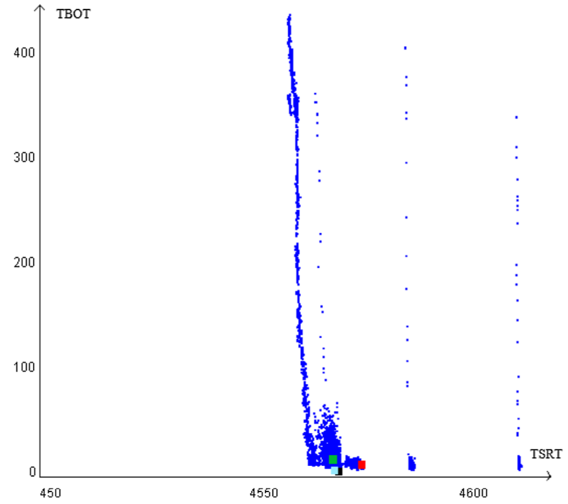


Figure 7. 2D-projection of Pareto Front approximation. Plane of total bay occupation time (TBOT) and total straddle carriers routing time (TSRT).

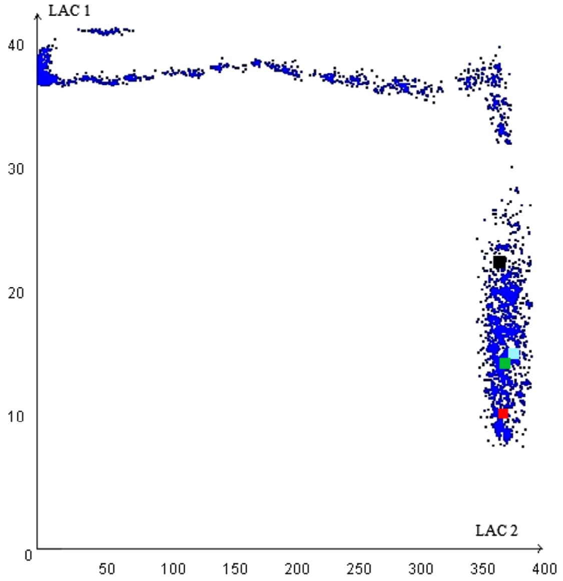


Figure 8. 2D-projection of Pareto Front approximation. Plane of location assignments Costs LAC1 and LAC2.

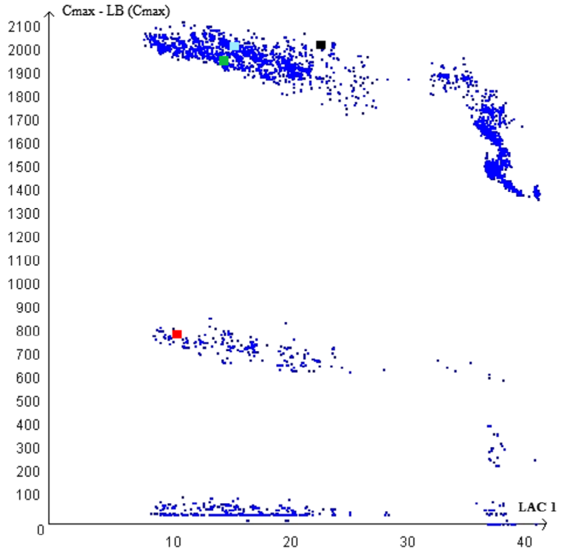


Figure 9. 2D-projection of Pareto Front approximation. Plane of C_{Max} and LAC_1.

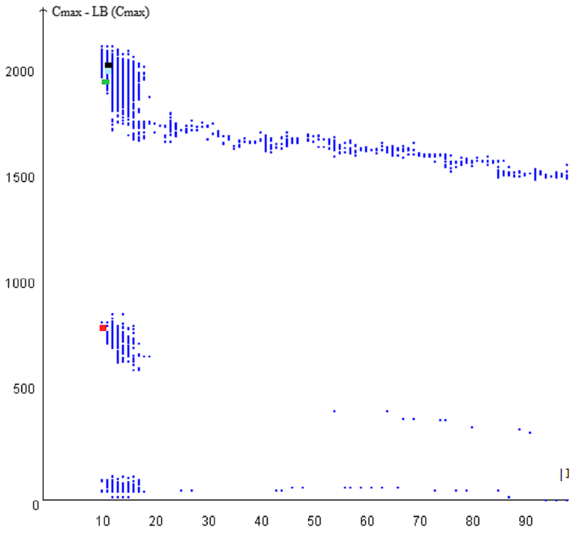


Figure 10 2D-projection of Pareto Front approximation. Plane of C_{Max} and $|B^{*}|$.

8.2. Election of potentially efficient solution

In Figures 7–10, we present 2D-projections of the Pareto Front. To elect solution from the set of approximated Pareto Front, four indicators of efficiency are presented in Section 7.11.

2D-projections of more than 5000 solutions are presented in Figures 7–10. Solutions proposed by indicators of efficiency are presented in the same figures, using specific colors.

Regarding others 2D-projections of the Pareto Front region, we select for this instance of the problem the solution proposed by the first indicator of efficiency. In Table 3, we present the best value which the gap reaches for every objective and also the worst values. We consider all the elements of approximated Pareto Front.

The objective values of solutions elected by indicators of efficiency have objective values presented in Table 4.

We evaluate the gap of every objective value for each solution proposed by indicators of efficiency. The gap is equal to the difference between objective value and its lower bound divided by the lower bound.

$$gap_k(S) = \frac{F_k(S) - LB_k}{LB_k} \quad \text{if } k < 8$$

$\text{gap}_8(S) = F_8(S)$ (different evaluation because $\text{LB}_8 = 0$).

$\text{gap}_k(S)$ Gap of solution S considering only objective k

$F_k(S)$ Objective value of solution S taking into account objective k

LB_k Lower Bound of objective k.

In Table 5, represent the gap of each solution each solution proposed by indicators of efficiency regarding every objective. Taking into account the approximated Pareto Front, the different lower bounds are reached efficiently for seven objectives from eight. For the last objective, which is the first location assignment cost and

Table 3. Best and worst objective gaps.

	C_{Max}	$\|V\|$	$\|B^*\|$	$\|P^*\|$	$\text{TSRT} = \sum_{v\in V} t_v$	$\text{TBOT} = \sum_{b\in B} t_b$	LAC_1	LAC_2
Best gap	0	0	0	0.21	0.09	0.02	7.8	0
Worst gap	0.08	0.17	14.09	7.43	0.21	62.06	40.93	399

Table 4. Objective values of elected solutions.

	IND	C_{Max}	$\|V\|$	$\|B^*\|$	$\|P^*\|$	$TSRT = \sum_{v\in V} t_v$	$\text{TBOT} = \sum_{b\in B} t_b$	LAC_1	LAC_2
Solution 1	4616	27930	19	11	141	526144	25762	10.88	374
Solution 2	4371	29096	18	12	140	519467	30130	14.88	376
Solution 3	5012	29155	18	12	143	520205	14612	15.62	382
Solution 4	2651	29163	18	12	140	520523	15120	23.02	371

Solution 1: Solution proposed by first indicator of efficiency (EI1).
Solution 2: Solution proposed by second indicator of efficiency (EI2).
Solution 3: Solution proposed by third efficiency indicator (EI3).
Solution 4: Solution proposed by fourth efficiency indicator (EI4).
IND: Index of the solution in Pareto list which represent the approximated Pareto Front. All the elements of Pareto list are presented in annexed part Objectives are denoted as presented in part 4.4.

Table 5. Objective gaps of elected solutions.

	IND	C_{Max}	$\|V\|$	$\|B^*\|$	$\|P^*\|$	$\sum_{v\in V} t_v$	$\sum_{b\in B} t_b$	LAC_1	LAC_2
Solution 1	4616	0.03	0.06	0	0.26	0.13	0.85	9.88	374
Solution 2	4371	0.07	0	0.09	0.25	0.12	1.17	13.88	376
Solution 3	5012	0.08	0	0.09	0.28	0.12	0.06	14.62	382
Solution 4	2651	0.08	0	0.09	0.25	0.12	0.06	23.02	371

IND index of solution in Pareto list given in annex 3 and annex 4 of the online supplementary material.

which represents the average distance between a container and the other containers to be delivered for the same client or at the same delivery date, the value of 7.8 *s* of routing time between two containers of same client is reached. Regarding practice, these values insure high quality of storage space.

For 2D-projection presented in Figures 7–10, we consider next representations of objective value projections:

Note that in Figures 9 and 10, the makespan is represented by its deviation from the lower bound: $C_{\text{Max}} - \text{LB}(C_{\text{Max}})$.

9. Conclusion

In this paper, we explore an NP-complete problem which has not been extensively studied: the multi-objective integrated problem of location assignment and straddle carrier scheduling in maritime container terminal at import. As we know, this work is the second study of this specific problem considering its integrated aspect and the first considering its multi-objective aspect.

In our approach, on the one hand, the integration of two optimization problems is theoretical guarantee of higher optimality. On the other hand, our solution proposes an 8-objective optimization process. It is also a new and efficient approach taking into account the real-world significance of the optimized objectives.

We developed a multi-objective Tabu Search algorithm (MOTSA) adapted to the studied problem. In the last part, we presented numerical results of MOTSA regarding a large real instance of 1000 containers and a total storage space capacity of 10000 containers.

Four indicators of efficiency are evaluated for each solution of the approximated Pareto Front in order to elect a limited set of potentially Pareto optimal solutions. The first and second indicators of efficiency are adapted to the combinatorial aspect of the problem, especially to the non-convexity of the Pareto Front Region. The third indicator is adapted to be easily used by operators in container terminal. The fourth indicator considers the two aspects.

2D-projections of the approximated Pareto Front are proposed to elect one element from solutions proposed by indicators of efficiency.

Seeing the approximated Pareto Front, the different lower bounds are reached efficiently taking into account the gap of every objective and the realistic needs of operators in container terminal for large instance of 1000 containers and 10000 free storage locations. The data instance is that of “Terminal de Normandie” in seaport of Le Havre.

Our Pareto Front approximation and elected solutions in particular, satisfy clearly the operator’s needs at Maritime Terminal using straddle carriers, theoretically as well as practically.

Acknowledgements

This work is co-financed by the European Union with the European Normandy Regional Development Fund (ERDF) and by the Normandy Regional Council.

Disclosure statement

No potential conflict of interest was reported by the authors.

References

Bish, E. K., Leong, T., Li, C., Ng, J. W. C., & Simchi-Levi, D. (2001). Analysis of a new vehicle scheduling and location problem. *Naval Research Logistics, 48*(5), 363–385.

Chen, L., Bostel, N., Dejax, P., Cai, J., & Xi, L. (2007). A tabu search algorithm for the integrated scheduling problem of container handling systems in a maritime terminal. *European Journal of Operational Research, 181*(1), 40–58.

Dawson, P., Jaeggy, D., Parks, G., Molina-Cristobal, A., & Clarkson, P. J. (2007). The development of Multi-thred Multi-objective Tabu Search Algorithm, Evolutionary Multi-Criterion Optimization. *Lecture Notes in Computer Science, 4403*, 242–256.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multi-objective genetic algorithm: NSGA-II. *Evolutionary Computation, IEEE Transactions, 6*(2), 182–197.

Gandibleux, X., & Feville, A. (2000). Tabu Search based procedure for solving the 0/1 multi-objective knapsack problem: The two objective case. *Journal of Heuristics, 6*(3), 361–383.

Giallombardo, G., Moccia, L., Salani, M., & Vacca, I. (2010). The tactical berth allocation problem with quay crane assignment and transshipment-related quadratic yard costs. In *Proceedings of the European Transport Conference (ETC)* (pp. 1–27).

Golias, M. N., Theofanis, S., & Boile, M. (2009). A bi-level formulation of the berth scheduling problem with variable vessel release dates to reduce port emissions. In *Proceeding of 2009 International Conference on Shipping, Ports, Logistics Management*. Korea, ISLC, Incheon: Inha University.

Hansen, M. (1997). Tabu search for multi-objective optimization: MOTS. In *MCDM*, Cape Town, South Africa.

Hansen, M. P. (2000). Tabu Search for multi-objective combinatorial optimization: Tamoco. *Control and Cybernetics, 29*(3), 799–818.

Jaeggi, D. M., Parks, G. T., Kipouros, T., & Clarkson, P. J. (2006). The development of multi-objective Tabu Search algorithm for continuous optimization problems. *European Journal of Operational Research, 185*(3), 1192–1212.

Jaegyy, D., Asselin-Miller, C., Parks, G., Kipouros, T., Bell, T., & Clarkson, J. (2004). Multi-objective parallel tabu search, parallel problem solving from nature. *Lecture Notes in Computer Science, 3242*, 732–741.

Kim, K. H. (1997). Evaluation of the number of re-handles in container yards. *Computers & Industrial Engineering, 32*(4), 701–711.

Kim, K. H., & Kim, K. Y. (1999). Routing straddle carriers for the loading operation of containers using a beam search algorithm. *Computers & Industrial Engineering, 36*(1), 109–136.

Kim, H. K., & Kim, H. B. (1999). Segregating space allocation models for containers inventories in port container terminals. *International Journal of Production Economics, 59*(1–3), 415–423.

Kim, K., Park, Y., & Ryu, K. (2000). Driving decision rules to locate export containers in container yards. *European Journal of Operational Research, 124*(1), 89–101.

Loukil, T., Teghem, J., & Tuyttens, D. (2005). Solving multi-objective production scheduling problems using meta-heuristics. *European Journal of Operational Research, 161*(1), 42–61.

Meersman, P. J. M., & Wagelmans, A. P. M. (2001). *Dynamic scheduling of handling equipment at automated container terminals* (Econometric Institute Report EI). Rotterdam: Erasmus University.

Nishimura, E., Imai, A., & Papadimitriou, S. (2005). Yard trailer routing at a maritime container terminal. *Transportation Research Part E: Logistics and Transportation Review, 41*(1), 53–76.

Tuyttens, D., Teghem, J., Fortemps, Ph., & Van Nieuwenhuyze, K. (2000). Performance of the MOSA method for the bi-criteria assignment problem. *Journal of Heuristics, 6*(3), 295–310.

Ulungu, E. L., Teghem, J., & Fortemps, P. H. (1999). MOSA method, a tool for solving multi-objective combinatorial optimization problems. *Journal of Multi-Criteria Decision Analysis, 8*(4), 221–236.

Vis IFA. (2000). *Planning and control concepts for material handling systems* (Thesis) (pp. 53–66).

A hybrid algorithm based on Particle Swarm Optimization and Simulated Annealing for electrical power transmission

Maria Zemzami[1], Norelislam Elhami[2], Mhamed Itmi[3] and Nabil Hmina[4]

[1,3]*LITIS – INSA – Normandy University, Rouen, 76000, France*
maria.zemzami@gmail.com
mitmi@insa-rouen.fr

[1,2,4]*LGS – ENSA – Ibn Tofail University, Kenitra, 14000, Morocco*
norelislam@outlook.com
hmina@univ-ibntofail.co.ma

Keywords: Optimization, metaheuristic, PSO, SA, hybrid algorithm.

1 Introduction

This work is about a novel hybrid algorithm that combines Particle Swarm Optimization (PSO) and Simulated Annealing (SA) algorithms. The basic idea behind using a hybrid model is improving the reliability of the obtained results from our first model, namely MPSO based on PSO algorithm, by adding SA algorithm which is quite popular for its powerful feature of effective escaping from the trap of local minima. Modified PSO model uses the concept of evolutionary neighborhoods associated to parallel computation, to overcome to the two essential disadvantages of PSO: high running time and premature convergence. MPSO is validated on a set of standard benchmark multimodal functions for which we obtained significant improvements, and also by improving the performance and reliability of the electricity pylon; the objective was to maximize resistance to load while reducing material usage and cost [1].

For this work, we provide the optimization of an electric power transmission material, giving specific consideration on material configuration and characteristics. The application problem studied is weight minimization of bars truss, by finding optimal cross-sectional areas of the truss members. The results obtained solving this optimization problem using the proposed hybrid algorithm is then compared with the results from ANSYS [2] first order conventional optimization technique.

For this model, a new concept of evolutionary neighborhoods is associated to the parallel model in order to improve the PSO performance. PSO is a stochastic metaheuristic based on population solutions. It searches for optimal solutions based on the concepts of cooperation and neighborhoods. Many variations and improvements of classical PSO version have been suggested by adapting its parameters, but good algorithm acceleration is required with a parallelization approach [3-4]. On the other hand, SA algorithm makes the search jump out of local optima due to its strong local-search ability. This hybrid approach makes full use of the global and local search optimization capability of both PSO and SA respectively and overcomes the weaknesses of each algorithm separately possesses.

In fact, by using PSO with SA, the advantages of both PSO (that has a strong global-search ability) and SA (that has a strong local- search ability) are combined. This is the basic idea of the HPSOSA.

Through application of SA to PSO, the proposed algorithm is capable of escaping from local optima and succeeds in converging into the global optima in the search space in a very good time consuming. The particularity of the approach consists to take advantage of the robustness of the PSO algorithm in choosing the right parameter setting, particularly the concept of dynamic neighborhood, in order to create diversity in research and in the sharing of information, the use of SA algorithm is important for a more optimal convergence. Besides the parallel computation that accelerates the calculations in order to have an optimal solution in an optimized computation time.

Experimental result shows that the developed hybrid PSO and SA algorithms can consistently produce the better result than MPSO and other algorithms.

Below the flowchart of the proposed model HPSOSA Figure1.

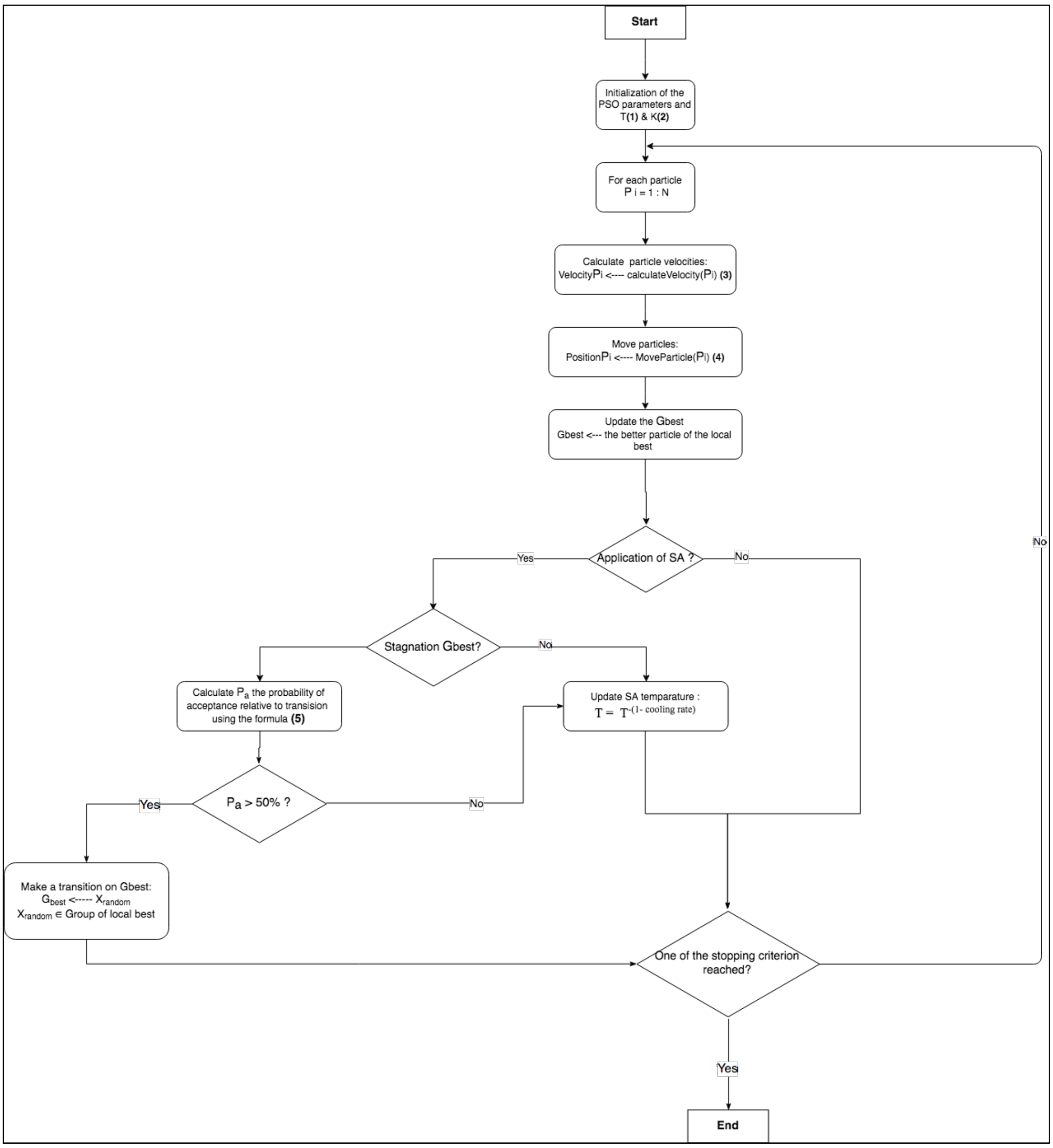

Start
Initialization of the PSO parameters and T(1) & K(2)
For each particle P i = 1 : N
Calculate particle velocities: VelocityPi <---- calculateVelocity(Pi) (3)
Move particles: PositionPi <---- MoveParticle(Pi) (4)
Update the Gbest Gbest <--- the better particle of the local best
Application of SA ?
Yes
No
Stagnation Gbest?
No
Calculate Pa the probability of acceptance relative to transision using the formula (5)
Update SA temparature : T = T-(1- cooling rate)
Pa > 50% ?
Yes
No
Make a transition on Gbest: Gbest <----- Xrandom Xrandom ∈ Group of local best
One of the stopping criterion reached?
No
Yes
End

Figure1: Hybrid MPSO-SA flowchart

2 References

[1] M. Zemzami, A. Elhami, A. Makhloufi, N. Elhami, M. Itmi, and N. Hmina (2017). « Applying a new parallelized version of PSO algorithm for electrical power transmission ». International Conference on Materials Engineering and Nanotechnology (ICMEN'17). Kuala Lumpur, Malizia. Indexed by Ei Compendex and Scopus.

[2] ANSYS Guide (2015). ANSYS Structural Analysis Guide.

[3] M. Zemzami, N.Elhami, M.Itmi and N.Hmina (2016). "Parallelization of the PSO algorithm on evolutionary neighborhoods". In: International Conference on Modeling, Optimization and Simulation (MOSIM'16). Montréal. Canada.

[4] M. Zemzami, N. Elhami, M. Itmi and N. Hmina, (2016)."A New Parallel Approach For The Exploitation Of The Search Space Based On PSO Algorithm". In: International Colloquium in Information Science and Technology (CIST'16). Tangier. Morocco. Indexed in Scopus.

3 Acknowledgements

This research is supported by "XTERM": Complex Systems, Territories Intelligence and Mobility, co-financed by the European Union with the European Regional Development Fund (ERDF) and Normandy Region.

Modelling the shortest Hamiltonian circuit problem in superimposed graphs with Distributed Constraint Optimization Problems

K. BOUAZZI[1], M. HAMMAMI[2] and S. BOUAMAMA[3]

1.COSMOS Laboratory, National School of Computer Science, University of Manouba (ENSI), Tunisia
Khaoula.bouazzi@ensi-uma.tn
2.COSMOS Laboratory, Higher Institute of management of Tunis, University of Tunis (ISG), Tunisia
moezhammami@gmail.com
3.FCIT, University of Jeddah KSA
Sbouamama@uj.edu.sa

Keywords : Superimposed graphs , Combinatorial optimization, Constraint satisfaction problems, Graph theory, Hamiltonian circuit problem.

1 Introduction

Graph theory is a wide field that is in constant evolution. Graphs are used to model many situations. Therefore, their applications are vast and also varied: in some branches of mathematics [1] (algebra, combinatory...), in computer science [2] (Computer networks [3], NoSQL [4], Wireless sensor networks [5]…), in operational research [6][7] (distribution transportation problems, scheduling problems…), cartography [8], chemistry [9], medicine [10]... A Hamiltonian circuit [11] in a graph is an ordering for a set of vertices such as every two (cyclically) consecutive vertices are joined by an edge. The problem of finding the shortest Hamiltonian circuit in a graph is one of the most famous problems in graph theory. For arbitrary graphs, the Hamiltonian circuit problem (HCP) is well known to be NP-complete. It is based on finding the shortest path. The most used algorithms for finding the shortest path is undoubtedly Dijkstra [12].

In the literature, special attention has been given to the problem of finding the shortest Hamiltonian circuit using exact methods such as branch and bound [13], A* algorithm [14], and dynamic programming [15]. These methods are effective for small size problems. For problems with more than two criteria or larger sizes, there is no exact effective approach, given the simultaneous difficulties of the NP-hard complexity and the multi-criteria framework of the problems [16][17]. Metaheuristics are essentially represented by local search methods such as simulated annealing, taboo research, and evolutionary algorithms, in particular genetic algorithms and evolution strategy algorithms [18]. Genetic algorithms are inspired by the mechanisms of evolution of living things and modern genetics, and are a powerful tool for optimization.

2 Approach description

Many graphs definitions are announced in literature. We will go along with the definition of J. A. BONDY and U. S. R. MURTY mentioned in their book [19]. A graph G is a pair *(X,A)* consisting of a nonempty set *X* of vertices and a set *A* of edges, joining the different elements of *X*.

In many cases, a normal graph cannot model some problems. Moreover, in some problems, vertices can exist in different categories, genres, and levels. However, simple graphs cannot guarantee this representation correctly, so we need a new model. Thus, we create a new graph called Superimposed graph (SG) [20].

A superposed graph (SG) is composed by an agglomeration of complete sub-graph connected by edges.

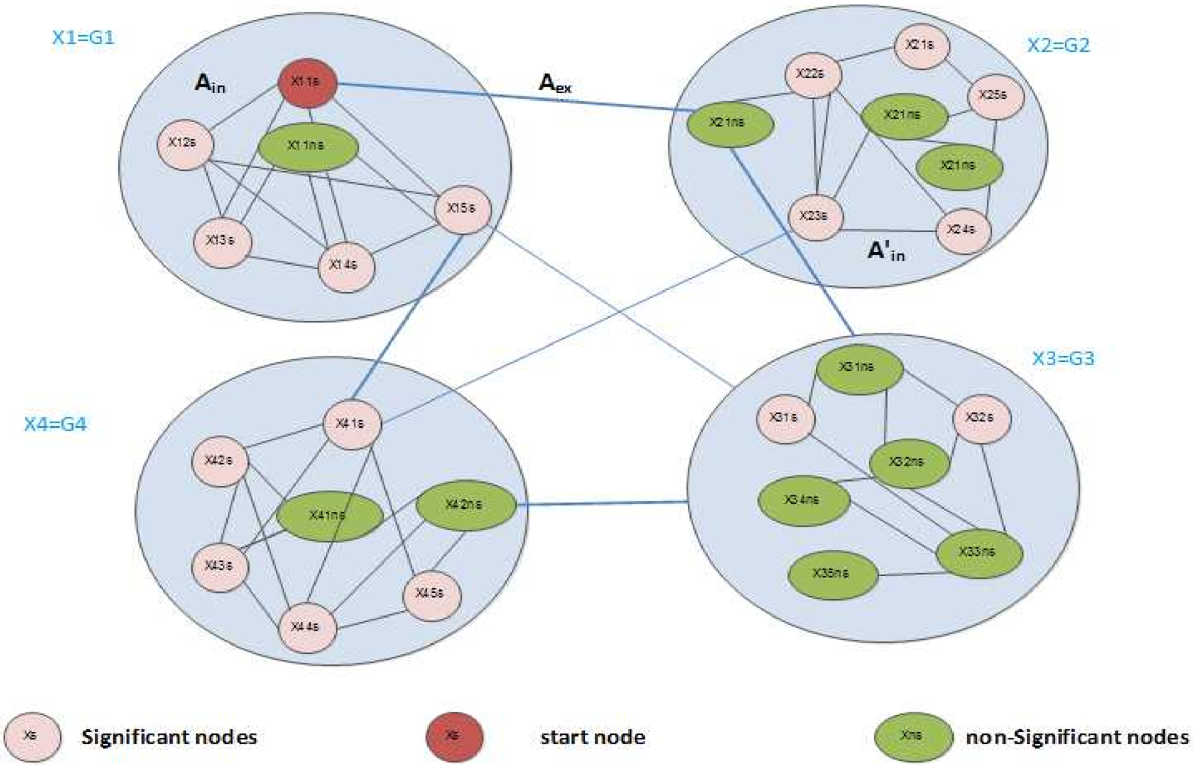


Figure 1 Superimposed graphs

Figure 1 represents a superimposed graph *G(X,A)*, each vertex *X* represents a **complete graph**. The edges A_{ex} represent **an external link** connecting the **complete graphs;** the edges A_{in} and A'_{in} represent **tow internal edges** connecting two vertices belonging to the same sub-graph. X_s represents significant vertices which must be visited in the tour and X_{ns} represents non-Significant vertices which could be visited.

Graphs are used to illustrate problems its purpose is to represent problems that are too numerous or complicated to be described adequately in a text. Moreover, graphs simplify the problem's resolution.

In optimization problems our main goals generally are to minimize the loss, maximize the gain and find the best solution. Therefore, to improve the problem's modeling; we should work on sets of networks (sub-graphs), avoiding consistently some vertices that make the problem more complicated or waste the time of execution. Normal graphs cannot provide that, all the vertices are similar with different links' weight.

In **superimposed Graphs** (SG) we don't have just similar vertices; we have two sets of vertices. The significant vertices X_s, the important ones, which must be visited and the non-significant vertices X_{ns} which could be visited if it upgrades the solution's research.

Moreover, in the SG case the edges also are divided into two sets, **internal** edges connecting the vertices of each network (A_{in}) and **external** edges (A_{ex}) which are connected vertices from different networks (sub-graphs). These partitions can decrease the flow's intricacy.

As it was mentioned previously, superimposed graphs are very important to model problems and simplify the way to find solutions. It minimizes the complexity and the waste of time. SG can be considered a very relevant addition in the graph theory.

Finding the shortest Hamilton circuit was a major problem in the field of multimodal transportation systems [21].The multimodal transport is defined by the combination of two or more transportation modes to move passengers or goods from an origin to a predetermined destination where a change of transportation mode is necessary. The multimodal transportation network studies have been used in many problems such as; planning networks, shortest paths, urban with maritime or airline centers, environmental issues, freight transport, transmission line, loading-unloading terminals, schedules, etc. For solving the shortest Hamilton circuit problem in the intricate multimodal transportation network, several approaches have been proposed.

Most of the works was modeled the transportation networks with a simple graph, but in our case we used the superimposed graphs.

Many real-world problems in artificial intelligence as well as in other fields of computer science and engineering can be efficiently modeled as constraint satisfaction problems (CSPs) [22] and solved using constraint programming techniques. Distributed Constraint Optimization Problems (DCOPs) [22] are problems where agents need to coordinate their value assignments, in a decentralized manner, to optimize their objective functions. Based on the formalism mentioned in the following papers [23] [24] and [25]; we presented the problem of finding the shortest Hamiltonian circuit in superimposed graphs with the DCOP .

For the resolution method we use a metaheuristics because the Hamiltonain circuit problem is NP-complete problem. We implement a distributed genetic algorithm [26] in JAVA. And for testing this approach we find that the transportation system of Guangzhou city of China [27] is the most suitable for our case, because of the heterogeneity of the transportation modes. So we attempted the algorithm to find the lowest Hamiltonian circuit on the areas of Guangzhou china. In our example, the vertices presented the modes of transport such as metro, bus, bicycle, and cars. The links presented the cost between every tow vertices.The data of the transportation system which is used to the testing phase were provided by a Laboratory of geographical information system GIS, in the School of Public Management, Guangdong University of Finance and Economics, Guangzhou China [27]. The data contains bus networks, metro network and walking links to the Guangzhou city. We collected all the information about the transportation system in one matrix to create a new benchmark for the superimposed graphs in multimodal networks.

3 Conclusion

In this paper, we have introduced the superimposed graphs which were used to find the shortest Hamiltonian circuit in a multimodal transportation system. Using a genetic algorithm we find that superimposed graphs are more efficient than normal graphs to solve problems such as finding the shortest Hamiltonian circuit in the multimodal transportation system.

On the basics of the concepts presented in this paper, a comparative work by other approaches is continuing and will be presented in future works. Experiments will be needed to verify the robustness of the superimposed graphs modelling to solve the Hamiltonian circuit problem using other metaheuristics.

References

1.DEHMER, Matthias and EMMERT-STREIB, Frank. *Quantitative Graph Theory: Mathematical Foundations and Applications*. CRC press, 2014.

2.DEO, Narsingh. *Graph theory with applications to engineering and computer science*. Courier Dover Publications, 2017.

3.BHARATH-KUMAR, Kadaba and JAFFE, J. "Routing to multiple destinations in computer networks". *IEEE Transactions on communications*. 1983, vol 31, p. 343–351.

4.ZHENG, Haifeng, YANG, Feng, TIAN, Xiaohua, GAN, Xiaoying, WANG, Xinbing and XIAO, Shilin. "Data gathering with compressive sensing in wireless sensor networks: A random walk based approach". *IEEE Transactions on Parallel and Distributed Systems*. 2015, vol 26, p. 35–44.

5.GADEPALLY, Vijay, BOLEWSKI, Jake, HOOK, Dan, HUTCHISON, Dylan, MILLER, Ben and KEPNER, Jeremy. *Graphulo: Linear algebra graph kernels for NoSQL databases*. 2015.p. 822–830.

6.IDRI, Abdelfattah, OUKARFI, Mariyem, BOULMAKOUL, Azedine, ZEITOUNI, Karine and MASRI, Ali. "A distributed approach for shortest path algorithm in dynamic multimodal transportation networks". *Transportation Research Procedia*. 2017, vol 27, p. 294–300.

7.NAGHSH, Zahra and VALAEE, Shahrokh. *GenS: A new conflict-free link scheduler for next generation of wireless systems*. 2017.p. 1–5.

8.HU, Yifan, SHI, Lei and LIU, Qingsong. "A coloring algorithm for disambiguating graph and map drawings". *IEEE Transactions on Visualization and Computer Graphics*. 2018.

9.JANEZIC, Dusanka, MILICEVIC, Ante, NIKOLIC, Sonja and TRINAJSTIC, Nenad. *Graph-theoretical matrices in*

chemistry. CRC Press, 2015.
10. BRAUN, Urs, MULDOON, Sarah F and BASSETT, Danielle S. "On human brain networks in health and disease". *eLS*. 2015.
11.VAN AARDT, Susan A, FRICK, Marietjie, OELLERMANN, Ortrud R and DE WET, Johan. "Global cycle properties in locally connected, locally traceable and locally hamiltonian graphs". *Discrete Applied Mathematics*. 2016, vol 205, p. 171–179.
12. BONDY, John Adrian, MURTY, Uppaluri Siva Ramach, RA and al., .*Graph theory with applications*. Springer, 2008.
13. WU, Er-Fei, JIN, Ye, BAO, Jin-Song y HU, Xiao-Feng. "A branch-and-bound algorithm for two-sided assembly line balancing". *The International Journal of Advanced Manufacturing Technology*. 2008, vol 39, no. 9-10, p. 1009–1015.
14. W.Zeng, R.Church. "Finding shortest paths on real road network". *International Journal of Geographical Information Science*. 2009 vol. 23, no 4.
15. WANG, Fei-Yue, ZHANG, Huaguang y LIU, Derong. "Adaptive dynamic programming: An introduction". *IEEE computational intelligence magazine*. 2009, vol 4, no. 2.
16. A.GUPTA, C. HENG, Y.ONG and al. "A generic framework for multi-criteria decision support in eco-friendly urban logistics systems*". Expert Systems with Applications*. 2017 vol. 71, p. 288-300 .
17. NESMACHNOW, Sergio. "An overview of metaheuristics: accurate and efficient methods for optimisation". *International Journal of Metaheuristics*. 2014, vol 3, p. 320–347.
18. SÖRENSEN, Kenneth y GLOVER, Fred W. *Encyclopedia of Operations Research and Management Science, Metaheuristics. 960-970 pp*. Springer US, 2013.
19. HOFFMAN, Karla L, PADBERG, Manfred y RINALDI, Giovanni. *Traveling salesman problem.* Springer, 2013, 1573–1578.
20. BOUAZZI, Khaoula, HAMMAMI, Moez and BOUAMAMA Sadok. "CSOP Framework for Lowest Hamiltonian Circuit in Superimposed Graph*".Proceeding of ISER international conference* ,2018 , vol 99 , p.39-43.
21. MNIF, Mouna y BOUAMAMA, Sadok. *A multi-objective formulation for multimodal transportation network's planning problems*. 2017.p. 144–149.
22.TSANG, Edward. *Foundations of constraint satisfaction: the classic text.* BoD–Books on Demand, 2014.
23. PETCU, Adrian and FALTINGS, Boi. *A scalablemethod for multiagentconstraintoptimization.* No. EPFL-REPORT-52705, 2005
24. ZIVAN, Roie, OKAMOTO, Steven and PELED, Hilla. "Explorativeanytime local search for distributedconstraintoptimization". *Artificial Intelligence*. 2014, vol 212, p. 1–26.
25. S LI, Shijie, NEGENBORN, Rudy R and LODEWIJKS, Gabriel. "Distributedconstraintoptimization for addressingvessel rotation planning problems". *Engineering Applications of Artificial Intelligence*. 2016, vol 48, p. 159–172.
26. BOUAMAMA, Sadok and GHEDIRA, Khaled. "A dynamic distributed double guided genetic algorithm for optimization and constraint reasoning". *International Journal of Computational Intelligence Research*. 2006, vol 2, p. 181–190.
27. CHEN, Shaopei, CLARAMUNT, Christophe and RAY, Cyril. "A spatio-temporal modelling approach for the study of the connectivity and accessibility of the Guangzhou metropolitan network". *Journal of Transport Geography*. 2014, vol 36, p. 12–23

Optimization of Vehicle Routing Problem in the Context of Reverse Logistics of Handling Containers in Closed Loop

K. Bouanane[1], Y. Benadada[2] and S. Marcotte[3]

1. Smart Systems Laboratory, Rabat IT Center ENSIAS, Mohammed V University in Rabat, Morocco
khaoula.bouanane@um5.ac.ma

2. Smart Systems Laboratory, Rabat IT Center ENSIAS, Mohammed V University in Rabat, Morocco
youssef.benadada@um5.ac.ma

3. Département Management et Technologie, École des Sciences et de la Gestion,
Université du Québec à Montréal (ESG-UQÀM)
Suzanne.marcotte@cirrelt.ca

Keywords: Reverse logistics, closed-loop, MDVRPSDPIR, Genetic Algorithm.

1 Introduction

The searches on the construction and the optimization of vehicle routing go to the direction of our industrial needs. The constructions of tours are both used for transportation of goods (direct logistics) and for the return of failing products, packaging, products of handling or at the end of life towards the productive system (reverse logistics). As for this work, it is particularly interested in the wooden pallet, the essential support of handling. Wooden pallet is used by all the manufacturers with satisfaction when it handles, it becomes then an embarrassment when it is empty. It appears that a system that, at a reasonable cost, unloads manufacturers for the daily management of empty pallets would be greatly appreciated. The system relies on management of park of pallets by a logistics service company dedicated to the pallet that allows its distribution, relocation to the desired place and collection, then repackaging and reinjection into a new supply chain. It is in this context that we are interested in the construction and the optimization of the tours taking into account, the direct flows (deliveries) and the indirect flows (pick-ups).

The industrial sector produces a more and more plentiful quantity of worn handling palettes, for which it becomes urgent to find clean solutions of elimination or recycling in an environmentally safe fashion. After its production, the pallet is transported, stored, distributed, stored again and then emptied before being re-transported and moved to a new supply chain. The palette is handled several times between the manufacturer and the end customer. In these handlings, it is necessary to add the movements during the various sorts, the repairs and the cleanings.

All these movements have a cost. The palette itself has a cost. Some manufacturers therefore prefer to rent pallets rather than to deal with these various and variable expenses. Others work with "lost" pallets for failing to manage their return. Others wish to receive reconditioning services and / or heat treatment pallets for their own fleets.

In recent years, the manufacturers think seriously to highlight the reverse logistics next to their classic supply chains. We propose, in this work, a structure of a pallet fleet management system where reverse logistics is integrated in the traditional supply chain, and it is in this context that we shall study a vehicle routing problem that takes into account the direct flow of pallets (deliveries) and indirect flows (pick-ups).

Part of the work is based on the establishment of a rather generic network where the pallets are delivered from one of the depots to several customers whose requests are known. In addition, customers express a demand for pallet collection in return. It is therefore a company manufacturing, renting/selling and repackaging pallets.

The company has a principal office and maintenance sites. Each site (principal or maintenance) contains a depot and a fleet of vehicles; it also contains a repackaging center with several workshops (sorting, repair and cleaning). With the exception of other sites, the principal site has a workshop manufacturing of new pallets and a pallet heat treatment unit.

This model corresponds to a Multi-Depot vehicle Routing Problem and with limited vehicle capacity, in which each customer can be receiver and deliverer at the same time, we are also interested in inventory management. It can therefore be classified in the category MDVRPSDPIR (Multi-Depots Vehicle Routing Problem with Simultaneous Delivery and Pick-up and Inventory Restrictions).

It is a closed-loop problem in witch constructs routes simultaneously for several vehicles from multiple depots to a set of customers, in a way that customers require simultaneous pick-up and delivery services. Another goal of this work is to find a solution at a minimal cost, which is in terms of total travel distance and number of vehicles, without violating the capacity constraints of the vehicles and the depots. Therefore, we developed two hybrid Genetic Algorithm (HGAs). In HGA1, initial solutions are generated randomly, while in HGA2, the Petal Heuristic and the Nearest Neighbor Heuristic are incorporated for initialization procedure. The numerical experiment will be performed using problem instances generated from 11 benchmark problems of Gillett and Johnson. Since we have not found other studies of the MDVRPSDPIR concerned, we will compare the results obtained by these algorithms with the optimal solution obtained by CPLEX for small instances.

Pricing & Lot Sizing problem in a hybrid manufacturing/ remanufacturing system with one-way substitution option

ZOUADI Tarik[1], YALAOUI Alice[2], REGHIOUI Mohamed[3]

1. Rabat Business School, International University of Rabat, Sala Al Jadida, Morocco
Tarik.zouadi@uir.ac.ma

2. LOSI (UMR CNRS ICD 6281), University of technology of Troyes, 12 Rue Marie Curie CS 42060, 10004 Cedex, France
yalaouia@utt.fr

3. National School of Business and Management, BP 1255, Tangier, Morocco
m.reghioui@gmail.com

Keywords: Lot-sizing, pricing, remanufacturing, memetic algorithm.

1 Introduction

Nowadays, manufacturers develop and enhance the reverse channels to handle and process the returns or collected used products (Wu 2012) [9]. Returns recovery can take a variety of forms: refurbishing, remanufacturing, repair, recycling, and cannibalization (King et al. 2006 [2], Thierry et al. 1995 [5]). The remanufacturing process consists of reconditioning and bringing returns to working status, which will allow their introduction on the market again [6]. Many companies consider the integration of remanufacturing process with their classical production activities as an opportunity to improve their profits, diversify their activities and confirm their sustainability [8]. However, in many cases, remanufactured products are downgraded in the market, therefore, remanufactured products are offered in many cases at an inferior price comparing to the new ones [3]. Integrating lot-sizing and pricing decisions allows companies to enhance their revenue and maximize their profit. González-Ramírez et al. [1] present a multi-item lot-sizing problem with pricing. The authors present a heuristic approach to solve large instances in moderate computational time. Ray et al. [4] present a profit-maximisation model for a hybrid manufacturing and remanufacturing system. The authors optimise selling price of new products and returns under production constraints using genetic algorithm.

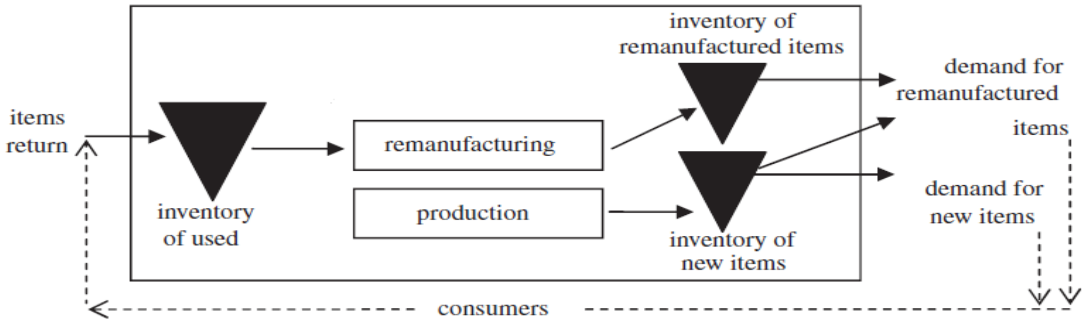


Figure 1: Flow scheme

In this study, we consider a single production line where both manufacturing and remanufacturing operations are performed (Figure 1). The products issued by the two processes are not assumed to be of the same quality. We consider that costumer is able to make difference between items issued from manufactured and remanufactured processes. One substitution option is considered, such that when the demand of remanufactured products is not fulfilled, the demand could be satisfied using the new products. We also assume that the remanufactured products have a lower price compared the new ones.

2 Model formulation

The proposed model is an extension of the hybrid manufacturing/remanufacturing model of Zouadi et al. (2015) [7]. The study integrates the pricing and lot sizing problem with one substitution way. We consider that demand D_t is sensitive to price P_t and defined as a linear function of price [1], Where u_t is the potential demand in the period t and b_t presents the sensitivity of demand in period t.

$$D_t(P_t)=u_t - b_t P_t$$

We consider the following notations. c_t^m, and c_t^r present the manufacturing and remanufacturing unit cost. K_t^m, and K_t^r denote the set up cost for manufacturing and remanufacturing. Finally,h_t^m, h_t^r and h_t^u are respectively the holding cost of new, remanufactured and returned products.

Decision variables:

P_t^r : Price of a remanufactured item to be sold in period *t*.
P_t^m : Price of a manufactured item to be sold in period *t*.
x_t^r : Number of products remanufactured at period *t*.
x_t^m : Number of products manufactured at period *t*.
x_t^s : Number of products substituted at period *t*.
δ_t^r : 0–1 Indicator variable for remanufacturing set-up at period *t*.
δ_t^m : 0–1 Indicator variable for manufacturing set-up at period *t*.
I_t^r : Inventory level of returns at the end of period *t*.
I_t^m : Inventory level of serviceable products at the end of period t.
I_t^u : Inventory level of returns at the end of period t.

The mathematical formulation is as follows:

$$\text{Max } \sum_{t=1}^{T}\Big[\big[(P_t^r D_t^r) + (P_t^m D_t^m)\big] - \big[(K_t^m \delta_t^m + c_t^m(x_t^m + x_t^s) + h_t^m I_t^m) + (K_t^r \delta_t^r + c_t^r x_t^r + h_t^r I_t^r + h_t^u I_t^u)\big]\Big] \qquad (1)$$

Subject to:

$$D_t^m(P_t^m)=u_t^m - b_t^m P_t^m \qquad \forall\, t = 1,2,3 \dots T \qquad (2)$$
$$D_t^r(P_t^r)=u_t^r - b_t^r P_t^r \qquad \forall\, t = 1,2,3 \dots T \qquad (3)$$
$$\mathrm{I}_{t-1}^{m} - \mathrm{x}_t^s + \mathrm{x}_t^m - D_t^m = \mathrm{I}_t^m \qquad \forall\, t = 1,2,3 \dots T \qquad (4)$$
$$\mathrm{I}_{t-1}^{r} + \mathrm{x}_t^s + \mathrm{x}_t^r - D_t^r = \mathrm{I}_t^r \qquad \forall\, t = 1,2,3 \dots T \qquad (5)$$
$$\mathrm{I}_{t-1}^{u} - \mathrm{x}_t^r + \mathrm{R}_t = \mathrm{I}_t^u \qquad \forall\, t = 1,2,3 \dots T \qquad (6)$$
$$\mathrm{x}_t^r \le \mathrm{M} \times \delta_t^r \qquad \forall\, t = 1,2,3 \dots T \qquad (7)$$
$$\mathrm{x}_t^m \le \mathrm{M} \times \delta_t^m \qquad \forall\, t = 1,2,3 \dots T \qquad (8)$$
$$\mathrm{x}_t^s \le \mathrm{M} \times \delta_t^m \qquad \forall\, t = 1,2,3 \dots T \qquad (9)$$
$$\delta_t^r,\ \delta_t^m \in \{0,1\} \qquad \forall\, t = 1,2,3 \dots T \qquad (10)$$
$$\mathrm{I}_t^r,\ \mathrm{I}_t^m,\ \mathrm{I}_t^u,\ \mathrm{x}_t^r,\ \mathrm{x}_t^m,\ \mathrm{x}_t^s \ge 0 \qquad \forall\, t = 1,2,3 \dots T \qquad (11)$$
$$\mathrm{I}_0^r = \mathrm{I}_0^m = \mathrm{I}_0^u = 0 \qquad (12)$$

The objective function (1) maximizes the total profit by subtracting the total costs from the sales revenue. Constraints (4), (5) and (6) are the inventory flow conservation equations for new, returned and remanufactured products. Constraints (7), (8) and (9) ensure the cancellation of remanufactured, manufactured and substituted quantities for the periods without setup cost. Constraint (10) defines the binary value of the manufacturing, remanufacturing decisions in each period. Constraint (13) ensures the positivity of the inventory, manufacturing, remanufacturing and substitution quantities decisions. Constraint (14) ensures that the initial stocks are equal to zero.

3 Memetic based algorithm

Memetic algorithms are a population-based Meta heuristics. The algorithm procedure begins by generating randomly an initial offsprings population. Based on the generated population, a selection process chooses two parents to form via the crossover operator the next generation offsprings. A small mutation is set at each iteration to ensure the diversification aspect of the algorithm. In the memetic algorithm, a local search procedure is added to enhance the solution quality and to locate efficiently the local optimum. This procedure is run until a stopping criterion. The memetic algorithm processes are adapted to our case as following:

- **Solution representation:** We choose a binary two-dimension vector to present the solution. For each period; we have two binary decisions indicating the manufacturing and, remanufacturing occurrence in each period.
- **Offsprings selection:** Based on the initial population generated randomly, we combine two selected parents to form two children (offspring), while keeping the good features of parents.
- **Mutation:** at each iteration a mutation is set by inverting randomly one of the binary variables
- **Local search:** a path relinking procedure is used to enhance solutions quality
- **Aspiration criterion:** The maximum number of iterations and execution time are the stopping criterion used in this implementation.

The binary decisions of the generated offspring's are integrated in the mathematical model as decision variables, which is then solved by CPLEX to define the optimal pricing and manufacturing and remanufacturing strategy.

4 Numerical experiments & conclusion

The proposed model was tested on CPLEX Software optimizer; the obtained results were compared with those obtained memetic-based algorithm. The quality of the solutions depends mainly on the size of the instance, which explains the idea of using memetic algorithm to solve the problem. For all the instances, the proposed approaches are compared with the optimal value returned by CPLEX, but when CPLEX cannot prove optimality, we compare with the lower bound. The preliminary obtained results have proven the performance and the relevance of the proposed approaches. Further researches can be considered by integrating availability and purchasing price of returns, and different quality level for remanufactured products.

References

[1] González-Ramírez, R. G., Smith, N. R., & Askin, R. G. (2011). A heuristic approach for a multi-product capacitated lot-sizing problem with pricing. International Journal of Production Research, 49(4), 1173-1196

[2] King, A. M., S. C. Burgess, W. I. jomah, and C.A. McMahon. 2006. Reducing waste: repair, recondition, remanufacture or recycle?, Sustainable Development, Vol 14(4), 257–267.

[3] Pineyro, P., & Viera, O. (2010). The economic lot-sizing problem with remanufacturing and one-way substitution. International Journal of Production Economics, 124(2), 482-488.

[4] Ray, A., & Mondal, S. (2018). An optimisation approach for finding an optimal pricing in a hybrid manufacturing/remanufacturing system. International Journal of Industrial and Systems Engineering, 29(1), 1-18

[5] Thierry, M., M. Salomon, J. Nunnen, L. Wassenhove. 1995. Strategic issues in Product Recovery Management, California Management review, Vol. 37, nº 2, pp. 114-135.

[6] Zouadi, T., Yalaoui, A., & Reghioui, M. (2018). Hybrid manufacturing/remanufacturing lot-sizing and supplier selection with returns, under carbon emission constraint. International Journal of Production Research, 56(3), 1233-1248.

[7] Zouadi, T., A. Yalaoui, M. Reghioui, and K. E. EL Kadiri. 2015. Lot-Sizing For Production Planning In A Recovery System With Returns, RAIRO Operations Research, Vol 49 1. 123-142.

[8] Wei, J., & Zhao, J. (2015). Pricing and remanufacturing decisions in two competing supply chains. International Journal of Production Research, 53(1), 258-278.

[9] Wu, C. 2012. "Price and Service Competition between New and Remanufactured Products in a Two-echelon Supply Chain." International Journal of Production Economics 140: 496–507

Dynamic Simulated Annealing with Adaptive Neighborhood using Hidden Markov Model

M. Lalaoui[1] and A. El Afia[2]

1. National School of Computer Science and Systems Analysis, Mohammed V University Morocco
med.lalaoui@yahoo.com
1. National School of Computer Science and Systems Analysis, Mohammed V University Morocco
a.elafia@um5s.net.ma

Abstract The Simulated Annealing (SA) is a stochastic local search algorithm. Its efficiency involves the adaptation of the neighbourhood structure. In this paper, we integrate Hidden Markov Model (HMM) in SA to dynamically adapt the neighbourhood structure of the simulated annealing at each iteration, based on the history of the search. An experiments was performed on many benchmark functions and compared with others SA variants.

Keywords: Simulated Annealing, Hidden Markov Model, Heuristic.

1 Introduction

Since the publication of the first article related to the SA algorithm [1], researchers have tried to increase the rate of convergence and to reduce the execution time of simulated annealing. Researchers focused since, the first version of the simulated annealing algorithm described by [1], on two strategies in order to improve the performance of SA. The first strategy was the implementation of parallel simulated annealing [2, 3, 4]. The second one was about the optimization of cooling schedule and the adaptation of parameters.

In general, the SA can be seen as an iterative improvement process composed from three functions: generation, acceptance and cooling.. Many researchers done an extensive studies on the update and accept function also the algorithm’s parameters and only limited attention has been paid to the generate function. The generate function governs the convergence of SA. When the SA is applied to continuous optimization problems, the appropriate adjustment of the neighborhood range according to the landscape of the given problem is very important. Because It significantly effects the accuracy of the solutions. In general, The SA tends to get stuck into local minima when the neighborhood range is too small.
In the special case of the fast-simulated annealing (FSA) [5], the state variation Δx is generated by a Cauchy distribution whose probability distribution is given as:

$$p(\Delta x; T_n) = \frac{1}{\pi} \frac{T_n}{(\Delta x)^2 + T_n^2} \quad (1)$$

The Cauchy distribution allows the SA to escape from local minima easier than the normal distribution because of its flatter tails [6].
The new neighborhood solution is generated using the formula:

$$x_{new} = x_{old} + \mu \times T_n \times \tan(u) \quad (2)$$

u is a random vector generated from the uniform distribution $U(-\pi/2 , \pi/2)$, μ is the scale constant for tuning the probabilistic acceptance criteria also called the learning rate and T_n is the temperature at stage n.
The temperature schedule is conducted by the following equation:

$$T_n = \frac{T_0}{1+n} \quad (3)$$

Where T_0 the initial temperature.

This paper explores the Hidden Markov Model for a dynamic tuning of FSA during the run. The main idea is to adapt the convergence speed through the dynamic adjustment of μ parameter. The rest of the paper is organized as follows. Section 2 is devoted to literature review, section 3 describes our proposed method for FSA parameter tuning through Hidden Markov Model. Section 4 describes a performance analysis on several benchmark functions, finally section 5 presents some conclusions.

2 Literature review

The fast simulated annealing is the most used in the literature, but the major drawback of this cooling policy is that it does not take into account the state of the system during the search. Thus, it is difficult to adaptively modify the intensity of the search based on the difficulty of the problem. According to Battiti and al. [7], a better performance can be achieved by a self-analysis during the search.

Researchers integrate a learning technique into the neighborhood function. An adaptive approach to adjust the neighborhood range of SA for continuous optimization problems was proposed by Corana [8]. But It has been proved by Miki et al. 2002 [9] that this method is not better than the SA with the good neighborhood range. Miki et al, 2002 [9] tried to enhance the performance of SA by appropriately adjusting the neighborhood range according to the landscape of the given problem using the opposition based learning. In fact, the opposition based learning increases only the diversity of the candidate's solution by selection not only the random guess but also its opposite. Furthermore, Miki et al, 2006 [10] also proposed a SA based on an adaptive search vector. This method guides the search direction according to the landscape of the problem. This adaptive search vector is based on the Powell's [11] method which is a direct search algorithm. It was reported by [12] that Powell's method may fails to converge when problem dimension is greater than 30. In addition, the experiments performed by Miki et al, 2006 [10] does not exceeding ten dimension. A simulated annealing algorithm with a dynamic neighborhood size adjusted according to the temperature parameter was proposed by Yao, 2007 [5]. The efficiency of this method is depending on a good adjustment of the temperature parameter. Thus a tuning for each problem will still necessary.

Machine learning methods proved their efficiency to predict with high accuracy. Those methods have been applied to adapt metaheuristic, which can be done by an off-line or online approaches. The offline one implies a configuration of features before executing the algorithm while the online adjusts the algorithm features during the run-time. In this paper, we apply the Hidden Markov Model to predict the best cooling law parameter. The Hidden Markov Model (HMM) [13] is a probabilistic model. It has been applied in many fields where information is not immediately observable but depend on other observable data. HHMs are the most successful approach in speech recognition, they have been successfully applied to the whole spectrum of recognition tasks. HMMs success is due to ability to deal with the variability by means of stochastic modeling. In addition, the HMM was successfully used to enhance the behavior of metaheuristics by estimating their best configuration [14, 15, 16, 17].

The main idea behind our approach is to control the behavior simulated annealing at each temperature stage by predicting the optimal geometric cooling law parameter according to the history of the run.

3 Proposed approach

An hybridization with the HMM was adopted to enhance the performance of FSA. During the run, the hidden Markov model performs classification based on observable sequence generated from a set of rules. This sequence allows the model to guess the hidden state which can be an exploration, exploitation, or Re-annealing to escape from a local minimum (Figure. 1).

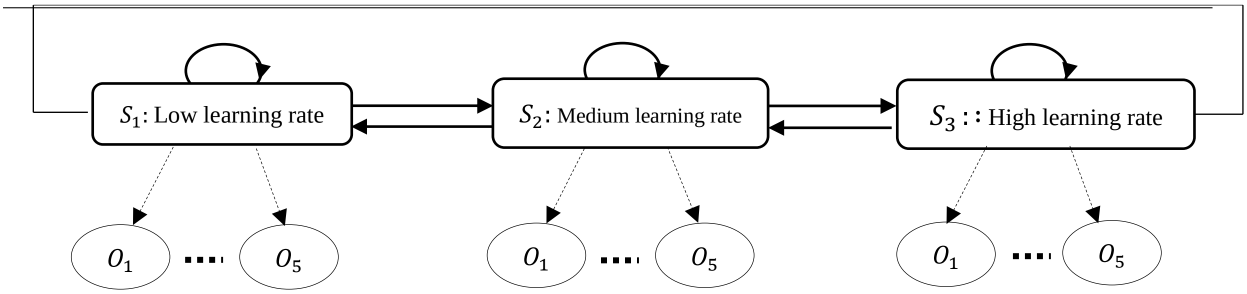


Fig. 1. Markov chain for simulated annealing algorithm

The Hidden Markov Model can be defined as 5-tuple (S, O, A, B, π^0) where:

- S= $\{S_1, S_2, S_3\}$ is the set of hidden states, which is respectively: low learning rate ($\mu = 0.2$), medium ($\mu = 0.5$), learning rate and high learning rate ($\mu = 0.8$).
- $O = (O_1, O_2 , \dots, O_5)$ is the set of the observation per state.
- A $= (a_{ij})$ is the transition probability matrix, where a_{ij} is the probability that the state at time $t + 1$ is S_j , is given when the state at time t is S_i
- $\pi^0 = (\pi_1^0, \pi_2^0, \pi_3^0)$ is the initial probability, where π_i^0 is the probability of being in the state S_i.
- $B = (b_{it})$ are the observation probabilities, where b_{it} is the probability of observing O_t in state S_i. This observations matrix B of hidden Markov model is estimated at early stage by Maximum Likelihood Estimation (MLE).

To estimate the observable matrix of HMM model, we generate a sequence of state containing the cooling schedule having the minimum variance in the cost $V(T)$ at equilibrium [14] for each stage of temperature (Algorithm 1). This measure of variance in cost is defined as: $V(T,f) = \langle V^2(\mathrm{T})\rangle - \langle V(T)\rangle^2$, where f is the cost function of the stationary distribution and $\langle f(T)\rangle$ is the expected cost in equilibrium for N last best cost value defined as :

$$\langle f(T)\rangle = \sum_{x_{F-N} \le x \le x_F} f(x) \frac{\exp\left(-\frac{f(x)}{T}\right)}{\sum_{x_{F-N} \le x \le x_F} \exp\left(-\frac{f(x)}{T}\right)} \tag{4}$$

The next step consists of generating the observations by the following algorithm:

Algorithm 1: Generate_Observation

```
Input:  $T, S_1, S_2, S_3$, $\{f(x), x_{F-N} \le x \le x_F\}$
Output: O Current Observation
 For i=1 to 3 do $V_i = V(S_i(T), f)$ End
     temp ← $V_1$
     For i=1 to 2 do
         If temp ≤ $V_{i+1}$ then O=i
                        else temp ← $V_{i+1}$ and $O$ ←i+1
     End
Return O
```

The main purpose of our model is to estimate states S that best explains the observation sequence O. Given the observation sequence $O = O_1 O_2 \dots O_{\mathrm{T}}$ and a model $\lambda = (A, B, \pi)$. Firstly, we estimate the transition and emission probabilities from the first sequence of observation using a supervised training. In which we count frequencies of transmissions and emission of the model:

Algorithm 2: MLE

```
Input:  $O = (O_1\ O_2 \dots O_{\mathrm{T}})$
Output:  A=$(a_{ij})$,B =$(b_{it})$
For i = 1 to T-1 do    $a_{O_iO_{i+1}} = a_{O_iO_{i+1}} + 1$      End
For i = 1 to T   do    $b_{O_iO_i} = b_{O_iO_i} + 1$           End
For i = 1 to 3   do    $A_i = \sum_{j=1}^{3} a_{ij}$ and  $B_i = \sum_{j=1}^{T} b_{ij}$      End
For $i = 1$ to 3 do
     For j=1 to 3   do $a_{ij} = a_{ij}/A_i$ End
     For t=1  to $T$ do $b_{it} = b_{it}/B_i$  End
 End
Return $A$
```

3.1 Viterbi algorithm

Then we use the Viterbi to select the corresponding state sequence $Q = q_1q_2 \dots q_T$ that best explains the observations, secondly, the Baum Welch adjust the model parameters $\lambda = (A, B, \pi)$ to maximize $P(O|\lambda)$, i.e., the probability of the observation sequence given the model. The Baum–Welch algorithm is used to adjust the parameters of the HMM. This training step is based on Forward-Backward algorithm.

Algorithm 3: Viterbi

```
Input: S= $(s_1, s_2, s_3)$ $O = (O_1\ O_2 \dots O_T)$, A = $(a_{ij})$  , B = $(b_{it})$ ,   $\pi^0 = (\pi_1^0, \pi_2^0, \pi_3^0)$
Output: $s^* = (s_1^*, s_2^*, s_3^*)$ the most probable sequence of states
For i = $1\ to\ 3$ do $\delta_1(i) = b_i(o_1)\pi_i$ and $\psi_1(i) = 0$    End    {Initalization}
 For t = 2 to  T do
    For j=1 to 3  do
          $\delta_t(j) = max_{i=1}^{3}[\delta_{t-1}(i)a_{ij}b_j(o_t)]$
          $\psi_t(j) = argmax_{i=1}^{3}\ [\delta_{t-1}(i)a_{ij}]$
     End
 End
 $P_{max} = max_{i=1}^{3}\ [\delta_T(i)]$
 $s_{10}^* = argmax_{i=1}^{3}[\delta_T(i)]$
 For t= $T - 1$ to 1 do $\boldsymbol{s_t^*} = \psi_{t+1}(s_{t+1}^*)$    End
 Return $\boldsymbol{s}^*$
```

3.2 Baum Welch algorithm

The Baum–Welch algorithm is used to adjust the parameters of the HMM. This training step is based on Forward-Backward algorithm.

- *Forward algorithm*

The first algorithm used by the Baum-Welch algorithm is the Forward algorithm. This algorithm returns the forward variable $\alpha_j(\mathrm{t})$ defined as the probability of the partial observation sequence until time t, with state S_j at time t,

$$\alpha_j(\mathrm{t}) = P\big(O_1 O_1 \dots O_t, q_t = S_j\big|\lambda\big) \quad (5)$$

We define $P(O|\lambda)$ as the probability of the observation sequence given the model λ.

Algorithm 4: Forward

```
Input:S=($s_1, s_2, s_3$),O=($O_1\ O_2 \dots O_{\mathrm{T}}$), A =$(a_{ij})$, B =($b_{it}$),$\pi^0 = (\pi_1^0, \pi_2^0, \pi_3^0)$
Output : $\alpha = (\alpha_{t+1}(j))$ , $P(O|\lambda)$
For  $i = 1\ to\ 3$ do $\alpha_i(i) = \pi_i b_{i1}$ End
For $t = 1$ to T-1  do
          For $j = 1$ to 3 do $\alpha_{t+1}(j) = \left(\sum_{i=1}^{3} \alpha_t(i)a_{ij}\right)b_{jt+1}$  End
End
 $\boldsymbol{P(O|\lambda)} = \sum_{i=1}^{3} \alpha_T(i)$
Return $\alpha, P(O|\lambda)$
```

- *Backward algorithm*

The second algorithm used by Baum-Welch Backward. This algorithm calculates the backward variable $\beta_i(t)$ defined as the probability of the partial observation sequence after time t, given state S_i :

$$\beta_i(t) = P(O_{t+1}O_{t+2} \dots O_T | q_t = S_i , \lambda) \quad (6)$$

The Baum-Welch is then used to re-estimate the parameters of the model $\boldsymbol{\lambda}$, which maximizes the probability of the observation sequence.

Algorithm 5: Backward

```
Input : S=( $s_1, s_2, s_3$), O=( $O_1\ O_2 \dots O_T$), A=$(a_{ij})$, B=( $b_{it}$), $\pi^0 = (\pi_1^0, \pi_2^0, \pi_3^0)$
Output : $\beta = \beta_t(i)$ the probability of the partial observation sequence
For $i = 1$ to 3 do $\beta_T(i) = 1$ End
For $t = T - 1$ to 1 do
        For $i = 1$ to 3 do $\beta_t(i) = \sum_{j=1}^{3} a_{ij}\beta_{t+1}(j)b_{jt+1}$ End
End

Return $\beta$
```

In the following we will implement a variant simulated annealing based on hidden Markov models. The interest behind hybridization and simulated annealing with the HMM is improved simulated annealing performance. This approach will allow the SA to overlap between slow and rapid cooling, this by changing the cooling schedule. Thus, the simulated annealing algorithm hybridization with HMM using Baum Welch and Viterbi algorithms is presented as follows:

Algorithm 3: HMM-SA algorithm

```
Data: The objective function $f$
Initialization : O : Empty sequence of observation, $T_0$: initial temperature , $T_f$ :final temperature , starting point $x \leftarrow x_0$ ,
cmp ← 1, u is a Random vector from the uniform distribution
Repeat
       Repeat
            $x_{new} = x_{old} + \mu \times T_n \times \tan(u)$
            If   $f(x_{new}) - f(x) \le 0$ then   $x \leftarrow x_{new}$
                                  else
                                   $P(x \leftarrow x\_new) = \exp\left(-\frac{f(x_{new}) - f(x)}{T}\right)$
            End
       Until equmbrium is approached sufficiently closely at $T_n$
       $O_n$ ← Generate-Observation$(T_n,\ f(x)_{x\_best_{F-9} \le x \le x\_best_F})$
 If $cmp \le 10$ then
                    $O \leftarrow [O, O_i]$
                    $[A, B] \leftarrow$ MLE$(O)$
                    state←Viterbi$(O, A, B)$
                    cmp ←cmp+1
               else
                    $O \leftarrow [O_{n-9}, \dots, O_n]$
                    [A, B] ←Baum-Welch(O,A,B)
                    state←Viterbi(O,A,B);
    End
   $\mu \leftarrow H_{state}$
   $T_n \leftarrow T_0/(1 + n)$
Until $T_{n+1} \le T_f$ indicating that the system is frozen
```

4 Experiments

Our experiments were designed to measure the effects of hybridization of HMM and SA and to show how our approach can improve the solution quality, we have chosen ten benchmarks selected from the literature. They are divided into two groups, unimodal functions with no local minimum except the global one and multimodal functions with many local minima. f_{min} is the known optima.

Name	Function Formula	Global Minimum
Six-Hump Camel	$f_1(\boldsymbol{x}) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2$	-1.0316
Rastrigin	$f_2\,(x) = \sum_{i=1}^{D} [x_i^2 - 10\cos(2\pi x_i) + 10]$	0
De Jong	$f_3(x) = \sum_{i=1}^{D} x_i^2$	0
Schuffer N°4	$f_4(x) = 0.5 + \frac{\cos^2(\sin(\lvert x_1^2 - x_2^2\rvert)) - 0.5}{[1 + 0.001(x_1^2 + x_2^2)]^2}$	0.292579
Colville	$f_5(x) = 100(x_1^2 - x_2)^2 + (x_1 - 1)^2 + (x_3 - 1)^2 + 90(x_3^2 - x_4)^2 + 10.1((x_2 - 1)^2 + (x_4 - 1)^2) + 19.8(x_2 - 1)(x_4 - 1)$	0
Levy N° 13	$f_6(x) = sin^2(3\pi x_1) + (x_1 - 1)^2[1 + sin^2(3\pi x_2)] + (x_2 - 1)^2[1 + sin^2(2\pi x_2)]$	0
Branin	$f_7(x) = (x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10$	0.397887
Quadric	$f_8\,(x) = \sum_{i=1}^{D}\left(\sum_{i=1}^{D} x_i\right)^2$	0
Shubert	$f_9(x) = \left(\sum_{i=1}^{5} i\,cos((i+1)x_1 + i)\right)\left(\sum_{i=1}^{5} i\,cos((i+1)x_2 + i)\right)$	$-186\,.7309$
Bukin	$f_{10}(x,y) = 100\sqrt{\lvert y - 0.01x^2\rvert} + 0.01\lvert x + 10\rvert$	0

Table 2. : Benchmark Functions

4.1 Experimental setup

The proposed hybridization of SA algorithm and HMM was coded in Scilab programming language and experiments were conducted on a PC with an Intel Core i7-5500U 2.40 GHz (4 CPUs) and 8 GB of RAM. For convenience, the SA algorithm with the vanilla version of fast simulated annealing, and the fuzzy fast simulated annealing [18].

These variants have been tested using the test functions presented above. Each function was tested over 50 trials. We eliminated the effects of other factors which play an important role in the performance of the algorithm, by choosing the same starting points for all methods (in each run) and their location was chosen to be far from the basins of attraction of global minima. Also, we have chosen the same initial acceptance probability and an identical length of the inner and outer loops.

The final temperature of the cooling process, T_{Final} have been taken close to zero. We fixed T_{Final} at 10^{-5}. The initial temperature T_0, have been calculated from mean energy rises Δf during the initialization. Before the start of the SA, the mean value of cost rises is estimated by a constant number of moves equal to 100. Then, initial temperature T_0 has been calculated using the following formula $\mathrm{T}_0 = \frac{-\Delta f}{\ln P_0}$ [19], where P_0 is the initial average probability of acceptance and is taken equal to 0.95. The length N of observed sequence was chosen equal to 10.

4.2 Numerical results

The computational results and statistical analyses are summarized in table 2. It provides the details of the results for the test functions. The overall best solution of the total 50 replications is shown in bold. HMM-FSA provide the best solution for the test functions f_1,. . , f_{10}.

In general, HMM-FSA algorithm overcome others variants in most of benchmark functions. In the most cases, our approach gives the best solution except for functions $f_1, f_2, f_4, f_5, f_6, \dots, f_{10}$.

Functions		**HMM-FSA**	**Fuzzy-FSA**	**FSA**
1	best	**-1.03E+00**	1.23E-03	4.67E+01
	mean	**-1.03E+00**	7.32E+02	1.35E+04
2	best	**5.99E-07**	2.61E-06	3.84E-01
	mean	1.27E-03	**2.75E-06**	5.39E+00
3	best	8.77E-12	**1.26E-12**	1.21E-04
	mean	2.48E-06	**1.34E-06**	4.73E-03
4	best	**5.00E-01**	5.56E-01	**5.00E-01**
	mean	**5.00E-01**	6.92E-01	**5.00E-01**
5	best	**1.22E-03**	6.23E+00	1.43E+02
	mean	**2.43E+02**	3.33E+05	3.92E+03
6	best	**3.60E-09**	1.18E-06	8.62E-06
	mean	**4.32E-05**	1.21E-02	4.91E-02
7	best	**3.61E-01**	8.45E-01	3.67E-01
	mean	**3.61E-01**	2.88E+00	3.67E-01
8	best	**1.08E-07**	1.17E-06	6.86E-05
	mean	**6.92E-04**	1.14E-03	5.72E-03
9	best	**-1.87E+02**	1.37E-09	**-1.87E+02**
	mean	**-1.87E+02**	3.72E-06	**-1.87E+02**
10	best	**1.45E-02**	2.03E-01	8.23E-02
	mean	**3.73E-01**	7.97E-01	6.48E-01

Table 3. Results comparisons between HMM-FSA, Fuzzy-SA and FSA

4.3 Comparison of convergence performance

For further insights into the convergence behavior our approach, HMM-SA method was compared to the three SA variants. The experiments were designed to measure the effects of the hybridization of FSA and HMM presented in the previous section. It was noticed that the HMM-SA can converge rapidly to the global minimum. The time gained in early stages can be used to converge to a better solution. This behavior is depicted in figure 2.

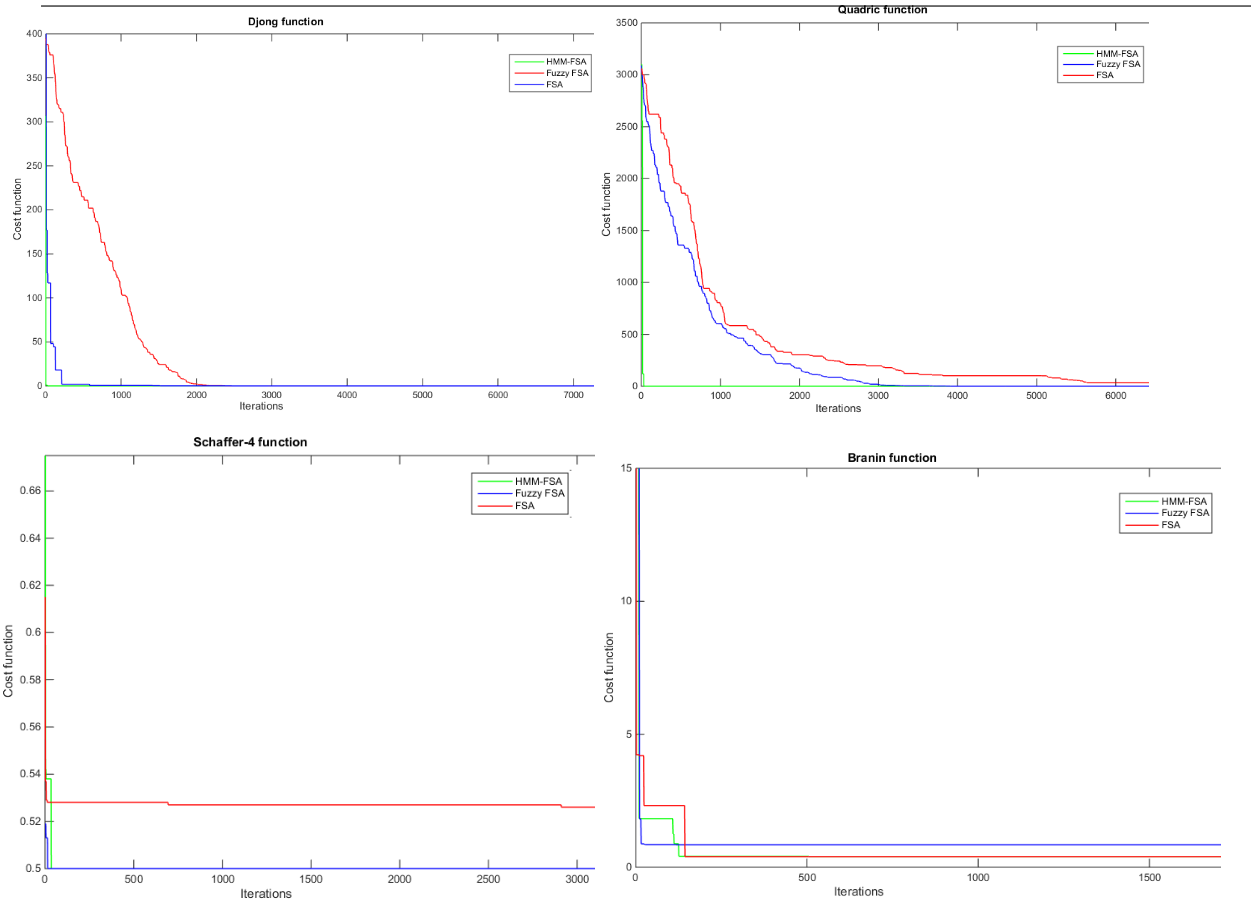


Fig. 2 Comparison of HMM-SA and SA variants

In the most cases, the HMM-FSA gives better results than other variants except for function f_3. The fast simulated annealing give a better solution over 50 runs for function f_3. Observed from figure 2 we can find that the cost function values of HMM-SA converges to the best solution. There are certain stages that others SA variant's outperforms HMM-SA. For example, the fuzzy simulated annealing convergence's speed is faster than HMM-SA for function Schuffer N°4 in the early. However, the convergence speed of HMM-SA is faster than others variants for functions f_3, f_7 and f_8.

4.4 Statistical analysis

The significance of the results has been evaluated using the Friedman's test [20, 21]. It is a non-parametric statistical test equivalent to the parametric ANOVA. The freedman's test Hypotheses are for formulated as follows:

H_0: Each ranking of the metaheuristics within each problem is equally likely, (i.e., there is no difference between them) so that for instance, the population medians are equal:

$$H_0 : [\mu_1 = \cdot\cdot\cdot = \mu_N] \qquad (7)$$

H_1: At least one of the metaheuristics has a different performance than at least one of the other metaheuristics:

$$H_1 : [\mu_1 ,\cdot\cdot\cdot, \mu_N \text{ not all equal}] \qquad (8)$$

In addition, we rank the results of the metaheuristic for each benchmark function, giving 1 to the best algorithm and 3 to the worst one. Let r(p_{ij}) be the rank of j^{th} algorithm in k algorithm on the i^{th} function of N benchmark functions, where k is equal to 6 and N is equal to 10 in our experiment. The average ranks of the algorithms were then computed by eq. 7, as shown in table 4.

$$R_j = \frac{1}{N}\sum_{i=1}^{N} \mathrm{r}(p_{ij}) \quad for\, j \in [1..6] \qquad (9)$$

The average ranks by themselves give a useful performance comparison. As depicted in table 4 the HMM-FSA ranks the first with the rank average of 1.1 followed by the Fuzzy FSA with the rank average of 2.3, the FSA ranks the third.

	HMM-FSA	Fuzzy FSA	FSA
f_1	1	2	3
f_2	1	2	3
f_3	2	1	3
f_4	1	3	2
f_5	1	2	3
f_6	1	2	3
f_7	1	3	2
f_8	1	2	3
f_9	1	3	1
f_{10}	1	3	2
Average rank (R_j)	**1.1**	**2.3**	**2.5**

Table 4: The rank for all algorithms in each benchmark function and the their average rank

The Freidman statistic was calculated by the following formula:

$$\chi_F^2 = \frac{10N}{k(1+k)}\left[\sum_{j=1}^{3} R_j^2 - \frac{k(k-1)^2}{4}\right] = 81.25 \quad (10)$$

Where the χ_F^2 statistic was distributed according to the F-distribution with $k-1=5$ and $(k-1)(N-1) = 18$ degrees of freedom. $\chi_F^2 = 81.25$ is greater than the critical values of $F(2, 18) = 3.55$ [29]. Thus, we reject the null hypothesis at the level of significance α=0.05. Then, we conclude that the performance of all algorithms is statistically different. We can proceed with a post hoc significant test to know if algorithm i and j are different.

5 Conclusion and future research

In this study, we proposed a dynamic simulated annealing with adaptive neighborhood using Hidden Markov Model. To test the performance of this approach, a number of benchmark functions have been applied. This approach allows to controls the neighborhood steps based on the history of the search. The HMM parameters are calculated and updated at each cooling step. The Viterbi algorithm is then used to classify the observed sequence and select the adequate learning value. The comparisons of the proposed approach with simulated annealing based on fuzzy control and the vanilla version of FSA, demonstrates that the simulated annealing based on HMM is able to find better solutions in reasonable time. Our approach is able to manage time by rapidly decreasing temperature and thus anticipating exploitation state, this lead to a better convergence. Future research may focus on the application of our method to some real optimization problems.

References

[1] S. Kirkpatrik (1983). *Optimization by simulated annealing. Science*, Vol. 220, No. 4598, 671–680.

[2] E. Aarts, F. de Bont, E. Habers, and P. van Laarhoven (1986). *Parallel Implementations of the Statistical Cooling Algorithm. Integration*, the VLSI Journal, Vol. 4, 209-238.

[3] P. Banerjee, M. Jones (1986). *A Parallel Simulated Annealing Algorithm for Standard Cell Placement on a Hypercube Computer*. Proceedings of the IEEE International Conference on Computer-Aided Design, 34-37.

[4] A. Casotto, F. Romeo, and A. Sangiovanni-Vincentelli (1986). *A Parallel Simulated Annealing Algorithm for the Placement of Macro-Cells*. Proceedings of the IEEE International Conference on Computer-Aided Design, 30-33.

[5] H.H. Szu and R.L. Hartley, 1987. Fast simulated annealing. Phys. Lett. A 122, 157–162.

[6] P. Melin , Olivas, F., Castillo, O., Valdez, F., Soria, J., and Valdez, M. (2013). Optimal design of fuzzy classification systems using PSO with dynamic parameter adaptation through fuzzy logic. Expert Systems with Applications, 40(8), 3196-3206.

[7] R. Battiti, M. Brunato (2008). *Reactive Search and Intelligent Optimization*, COMPUTER SCIENCE INTERFACES, Springer.

[8] A. Corana, M. Marchesi, C. Martini, and S. Ridella, 1987. “Minimizing Multimodal Functions of Continuous Variables with the Simulated Annealing Algorithm”, ACM Trans. on Mathematical Software, Vol. 13, No. 3, pp. 262-280.

[9] M. Miki, T. Hiroyasu, K. Ono, 2002. “Simulated Annealing with Advanced Adaptive Neighborhood” .In Second international workshop on Intelligent systems design and application, pages 113–118, Atlanta, GA, USA

[10] M. Miki, S. Hiwa, 2006. T. Hiroyasu. “Simulated Annealing using an Adaptive Search Vector, Cybernetics and Intelligent Systems.

[11] M.J.D. Powell, “An Efficient Method for Finding the Minimum of a Function of Several Variables without Calculating Derivatives,” Computer J., 7, pp. 155–162,

[12] R. T. Haftka and Z. Gurdal, “Elements of Structural Optimization”, Solid Mechanics And Its Applications, Volume 11 1992, Book, chap.4 ,p. 124, sec. 4.2.

[13] L. Rabiner (1989*). A tutorial on hidden markov models and selected applications in speech recognition*. Proceedings of the IEEE, 77(2), 257–286.

[14] O. Aoun, M Sarhani and A. El Afia (2016). *Investigation of hidden markov model for the tuning of metaheuristics in airline scheduling problems*. IFAC-PapersOnLine, 14th IFAC Symposium on Control in Transportation Systems, Turkey,Vol. 49, Issue 3 p 347–352.

[15] S. Bouzbita, A. El Afia and R. Faizi (2016). *A Novel Based Hidden Markov Model Approach for Controlling the ACS Evaporation Parameter*. The 5th International Conference on Multimedia Computing and Systems IEEE Conference, ICMCS'16, Marrakech, Morocco.

[16] M. Lalaoui, A. El Afia and R. Chiheb (2016). *Hidden Markov Model for a Self-Learning of Simulated Annealing Cooling Law*. The 5th International Conference on Multimedia Computing and Systems IEEE Conference, ICMCS'16, Marrakech, Morocco.

[17] O. Aoun, M. Sarhani, A. El Afia (2018). Hidden markov model classifier for the adaptive particle swarm *optimization*. In Recent developments of metaheuristics, 1–15.

[18] M. Lalaoui, A. El Afia and R. Chiheb (2018). Simulated Annealing with Adaptive Neighborhood using Fuzzy Logic Controller, The International Conference on Learning and Optimization Algorithms: Theory and Applications (LOPAL'2018), on-going for publication.

[19] P. Kouvelis, W.-C. Chiang, J.A. Fitzsimmons (1992). *Simulated Annealing Procedures for Machine Layout Problems* in the Presence of Zoning Constraints. Eur J Oper Res 57(22), 203–223.

[20] M. Friedman (1937). *The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance*. J Am Stat Assoc, 32(200):675–701.

[21] A. LaTorre, S Muelas, J M Peña (2015). *A comprehensive comparison of large-scale global optimizers*. Information Sciences 316, 517-549.

Fast Generation of Combinatorial Objects

Victor Parque[1,2] and Tomoyuki Miyashita[1]

[1] Department of Modern Mechanical Engineering,
Waseda University
3-4-1 Okubo, Shinjuku-ku, Tokyo, 169-8555, Japan
[2] Department of Mechatronics and Robotics,
Egypt-Japan University of Science and Technology (E-JUST)
Qesm Borg Al Arab, Alexandria Governorate 21934, Egypt
Corresponding Author: parque@aoni.waseda.jp, victor.parque@ejust.edu.eg

1 Introduction

Combinations are relevant representational mechanisms allowing to tackle optimization problems in various instances [1–12]. In this paper, we focus in combinations of n objects taken m at the time, or well-known as m out of n, and its efficient unranking mechanism.

In particular, the unranking of m out of n consists in generating the object $(x_1, x_2, ..., x_m)$ given an integer number $g \in [0, \binom{n}{m} - 1]$. Generating such combination is advantageous in the following problem scenarios

- in combinatorial instances in which the integer $g \in [0, \binom{n}{m} - 1]$ can be used as a succinct encoding of the combination element of size m out of n,
- in scenarios in which parallelization is able to speed up the sampling and evaluation of a subset of combination objects from the complete combinatorial space,
- in scenarios in which the evaluation of duplicated combinations is expensive and irrelevant, and the integer number g is a surrogate to identify and relate to $(x_1, x_2, ..., x_m)$.
- in scenarios in which generating all combinations is computationally expensive, specially for large $\binom{n}{m}$, and generating a small subset is rather preferable.

The unranking of combinations has been studied since the late seventies, in which Buckles and Lybanon [13] proposed the first approach, and a correction was proposed thirty years later by Crouse [14]. The first hardware realization was proposed by Butler and Sasao [15]. Basically, a fundamental approach for unranking combinations finds the maximal element x_m subject to $\binom{x_m}{m} \leqslant g$; then, finds the maximal x_{m-1} subject to $\binom{x_{m-1}}{m-1} \leqslant g - \binom{x_m}{m}$, and so on. However, since computing binomial coefficients is computationally expensive, various surrogate approaches were introduced [16–26].

Generally speaking, the efficiency of the conventional unranking algorithms depends on the value of n, which is restrictive for very large n. In a different approach, n-independent unranking is of special interest in the following scenarions:

- to allow further scalability for very large n and for cases in which m is much smaller than n,
- to allow adaptability when n is time-varying

Yet, n-independent unranking of combinations has received little attention in the literature. Kokosinskiński proposed UNRANKCOMB-E as the first algorithm with $O(m)$ time [19] and, Shimizu et. al. proposed an algorithm with $O(m^{3m+3})$ time [27]. Whereas, the UNRANKCOMB-E algorithm requires n^2 processors and $O(n^2)$ space, the algorithm proposed in Ref. [27] is restricted to sampling through Dynamic Programming. Also, the efficient parallelization of n-independent unranking is non-trivial due to requiring at least quadratic number of processors (which is detrimental for standard computing environments), and believed polynomial space requirements[3].

In this paper, to fill the above gaps, we capitalize the advantages of existing parallel computing to render n-independent unranking of combinations, and evaluate the experimental performance of a gradient-based approach for unranking. Our basic idea is to generate combination elements by solving minimization problems, through a parallelizable objective function.

[3] personal communications with Corresponding Author of [27].

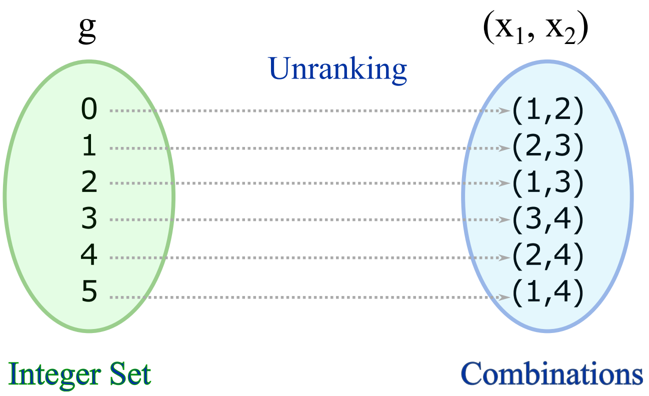


Fig. 1. Basic Idea of Unranking Combinations of m elements taken m at the time for $n = 4$ and $m = 2$. Left: the integer set in which $g \in [0, \binom{n}{m} - 1]$. Right: Combination tuples.

Our contributions are summarized as follows:

– We present exhaustive experiments showing the practical efficiency of an optimization-based unranking of combinations.
– The time for unranking combinations is efficient experimentally in our computationally allowable limits[4], showing the feasibility and efficiency to unrank combinations with n-independency.
– The time complexity shows a linear-like behaviour.

2 Proposed Approach

In this section we outline the basic idea of n-independent unranking of combinations based on minimization of a parallelizable log-concave function.

2.1 Preliminaries

Given positive integers n and m, and $[n] = \{1, 2, 3, ..., n\}$, a combination object is defined as the tuple $(x_1, x_2, x_3, ..., x_m)$ in which $x_i \in [n], i \in [m]$ and $n \geq m$.

For integer number g, the unranking algorithm consists in generating the combination object uniquely associated with the number g, with respect to some ordering, as follows:

$$\textbf{unranking} : g \rightarrow (x_1, x_2, x_3, ..., x_m) \tag{1}$$

, where

$$0 \leq g \leq \binom{n}{m} - 1$$

To exemplify the above definition, Fig. 1 shows the basic idea of an unranking instance for $n = 4$ and $m = 2$.

To encode combinations, we use the *revolving* door order starting with $(1, ..., m)$, and ending with $(1, ..., m-1, n)$ [22]; wherein combinations are recursively divided either including or avoiding the element n, and holding the following condition:

$$x_1 < x_2 < ... < x_m$$

The key motivation of using the revolving door order is due to the minimal change order and the *ordered cyclic sequence* since the Hamiltonian path in a polytope formed with $\binom{n}{m}$ vertices minimizes the distance between neighbor (close) combinations [23]. Thus, the above facts make the *revolving door* relevant to build sampling-based learning algorithms with *canonical* concepts [28].

[4] the numerical limits in our standard computing environment, 1.7977×10^{308}

Algorithm 1 Unranking Algorithm

```
procedure UNRANK($g, m$)
    Input $g$                                                    ▷ Rank Number
    Output $(x_1, x_2, ..., x_m)$                                ▷ Combination Object
    $x_m^o \leftarrow m$                                         ▷ Initial approximate solution
    for $i \leftarrow m$ downto 1 do
        $x_i \leftarrow 1 + \left\lfloor \text{Minimize } |J(x, i, g)| \text{ with } x \geqslant i \right\rfloor$    ▷ Solve a minimization problem
        $g \leftarrow \binom{x_i}{i} - g - 1$                    ▷ Binomial Coefficient
    end for
    return $(x_1, x_2, ..., x_m)$                                ▷ Combination object
end procedure
```

2.2 Minimization Problem

The basic idea in our proposed unranking algorithm is shown by Algorithm 1, in which the element x_i is generated by minimizing the cost function $|J(x, i, g)|$ for variable x and constraint $x \geq i$. Here, the log-concave objective function is as follows:

$$J(x, i, g) = \left[\sum_{p=1}^{i} \log_b \left(\frac{x - i}{p} + 1 \right) \right] - \log_b(g), \tag{2}$$

and

$$|J| \in x \in [i, +\infty) \tag{3}$$

where i denotes the index in the *for* loop of Algorithm 1 ($i = m$ at initial iteration), g denotes the integer number representing the combination, and the base b in $\log_b(.)$ (the larger the coefficient $\binom{n}{m}$ is, larger the b to handle very large numbers accurately). For standard computing environments, we use $b = 10$.

To find the minimal of $|J|$ in $x \in [i, +\infty)$, we used the terms of the Taylor Series up to the first order, as follows:

$$x_i^{k+1} = \begin{cases} x_i^k - \frac{J}{J'}, & \text{if } x_i^{k+1} \geqslant i \\ i, & \text{otherwise} \end{cases} \tag{4}$$

, where $i \in [m]$, k denotes the iteration number, and the subscript J' denotes the first derivative of the function J with respect to x_i^k.

$$J'(x, i) = \frac{1}{\ln(b)} \sum_{p=0}^{i-1} \frac{1}{(x - p)} \tag{5}$$

Due to the concavity of the function J in $x \in [i, +\infty)$, the initial solutions $x_i^o \in [i, +\infty)$ ensure convergence to the root of J, and are initialized as follows:

$$x_i^o = \begin{cases} m, & \text{if } i = m \\ x_{i+1}, & \text{otherwise} \end{cases} \tag{6}$$

The above is based on the revolving door ordering principle: $x_1 < x_2 < ... < x_m$, and the closeness of x_i to x_{i+1}, which ensures efficient convergence to the global optima.

As for termination criterion, we use the following criteria:

$$|J'| < \delta \tag{7}$$

and

$$|x^{k+1} - x^k|/|x^k| < \varepsilon \tag{8}$$

in which δ and ε are threshold tolerances to avoid division and sampling with very small numbers. Without loss of generality within the context of standard desktop environments, we use $\delta = \varepsilon = 10^{-8}$.

3 Computational Experiments

In order to evaluate the computational efficiency of the unranking algorithm, we used diverse scenarios which reflect our foci for future potential applications, as well the applicability in standard computing environments. We implemented Algorithm 1 in MATLAB, Intel i7 4930K @ 3.4GHz, Windows 8.1, 64 bit, wherein the computation of the subscript $\frac{J}{J'}$ in Eq. 4 is realized through a Graphics Processing Unit, GPU, GeForce GTX TITAN.

Thus, in line of the above, to ensure the rigourous evaluations, we measured the time involved in the following tasks:

- Parallelization of the log-concave objective function
- Time for unranking all combinations for different values of n and m.

In the following subsections, we describe our experimental settings and our observed insights.

3.1 Parallelization of the Objective Function

In order to evaluate the gains in efficiency between CPU and GPU realizations when computing the function $J(x, i, g)$ and $J'(x, i)$, we performed the following experiments:

- Comparison of efficiency between CPU and GPU implementations of the function $J(x, i, g)$ for equally distributed points of $i \in [1, 10^8]$.
- Comparison of efficiency between CPU and GPU implementations of the function $J'(x, i, g)$ for equally distributed points of $i \in [1, 10^8]$.

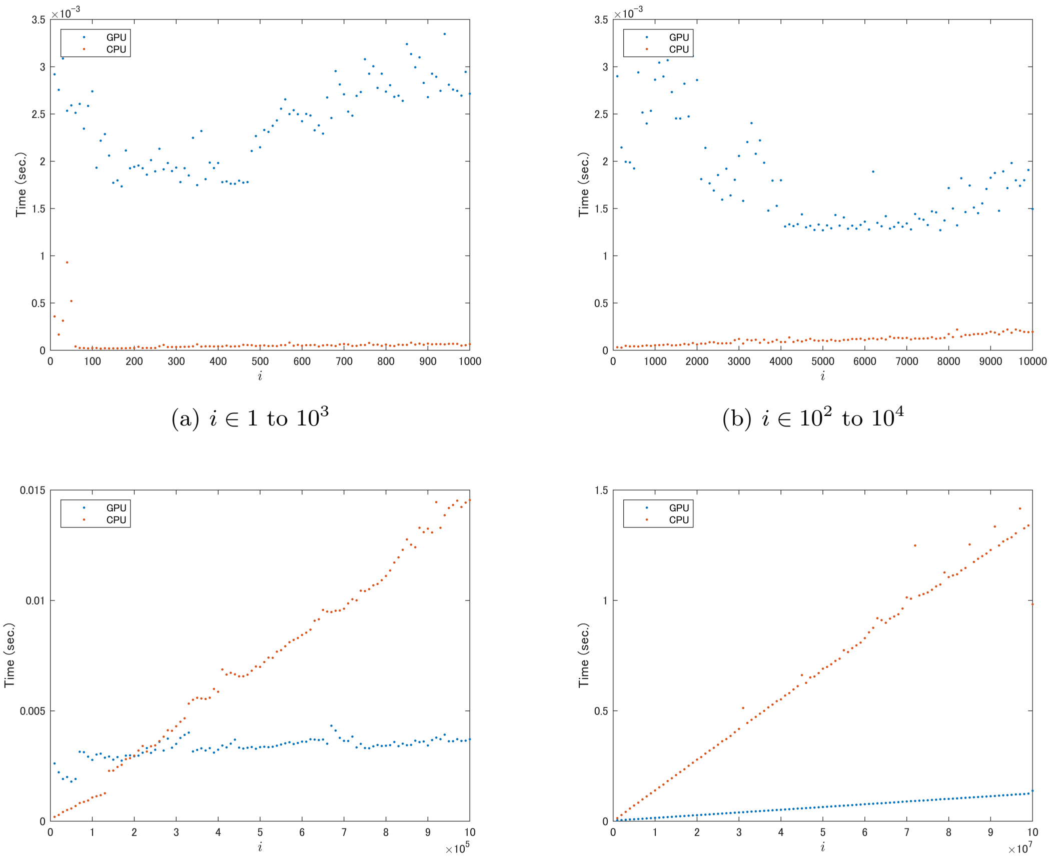
The above is done for $x = 10^8$, $g = 3$ and four discretizations of the interval $[1, 10^8]$. Note that the efficiency frontiers when computing the functions $J(x, i, g)$ and $J'(x, i)$ is dependent of i only, in which x and g are constants.

As for the CPU implementations, both the function $J(x, i, g)$ and $J'(x, i, g)$ use a conventional serial mechanism to reduce the sums. Thus, the theoretical time is expected to be $O(i)$ time by using a single processor.

As for the GPU implementations, both functions J and J' are computed in $O(\log i)$ time by using at most $O(i/\log i)$ processors (to ensure work-efficiency in parallel cores according to the Brent's Theorem[5]). The parallelization of the cost function J and J' use a *reduction* algorithm of i sums by multiple elements per thread in the CUDA-enabled GPU.

To evaluate the performance gains when using CPU and GPU realizations, Fig. 2 and Fig. 3 shown the time involved in computing $J(x, i, g)$ and $J'(x, i, g)$ for the above mentioned scenarios. By observing Fig. 2 and Fig. 3, we can confirm the following facts:

[5] assuming algebraic operations with numbers in $O(1)$



(a) $i \in 1$ to 10^3

(b) $i \in 10^2$ to 10^4

(c) $i \in 10^4$ to 10^6

(d) $i \in 10^6$ to 10^8

Fig. 2. Comparison in computational efficiency between CPU and GPU realizations when computing the function $\mathbf{J(x,i,g)}$ for $x = 10^8$, $g = 3$ and 100 equally distributed points of i.

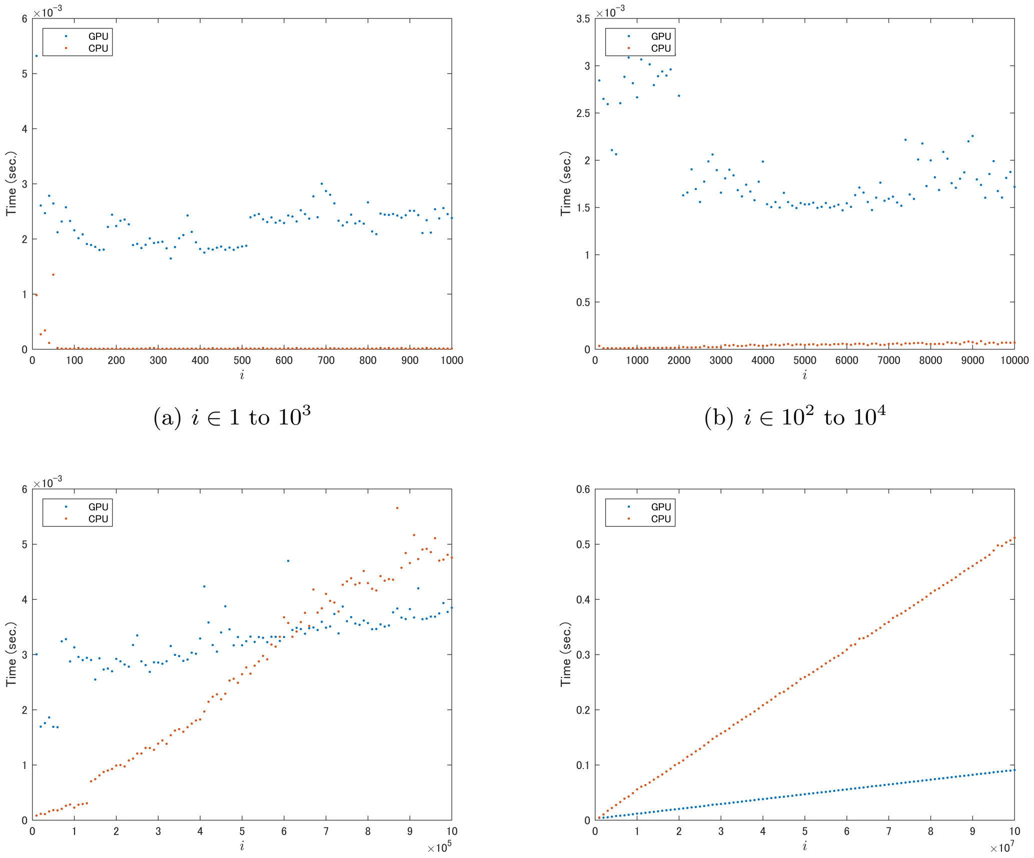
- Computing either $J(x,i,g)$ and $J'(x,i,g)$ takes less than a 6 miliseconds in either CPU and GPU for $i \in [1, 10^4]$.
- Computing either $J(x,i,g)$ ($J'(x,i,g)$) takes less than a 15 (6) miliseconds in either CPU and GPU for $i \in [10^4, 10^6]$.
- Computing either $J(x,i,g)$ ($J'(x,i,g)$) takes less than a 1.5 (0.6) seconds in either CPU and GPU for $i \in [10^6, 10^8]$.
- Using the GPU to compute $J(x,i,g)$ and $J'(x,i,g)$ is more advantageous for large values of i.
- The comparative advantages of using a GPU are positively increased in proportion to i increments.

The above observations show that the parallelization of the proposed log-concave objective function brings practical benefits in scenarios involving large values of i. This fact implies the potential benefits for scalability.

3.2 Unranking Time

In order to evaluate the efficiency of the proposed unranking algorithm, we used relevant values of numbers n and m within computable float numbers for binomial coefficients in a standard desktop environment, which in our case is represented as 1.7977×10^{308}.

Thus, to realize our aim of evaluating the proposed approach exhaustively, we generated combination objects associated with numbers *uniformly distributed* on a fixed range. For simplicity and without loss of generality, we used the following scenarios:



(a) $i \in 1$ to 10^3

(b) $i \in 10^2$ to 10^4

(c) $i \in 10^4$ to 10^6

(d) $i \in 10^6$ to 10^8

Fig. 3. Comparison in computational efficiency between CPU and GPU realizations when computing the function $J'(x, i)$ for $x = 10^8$, $g = 3$ and 100 equally distributed points of i.

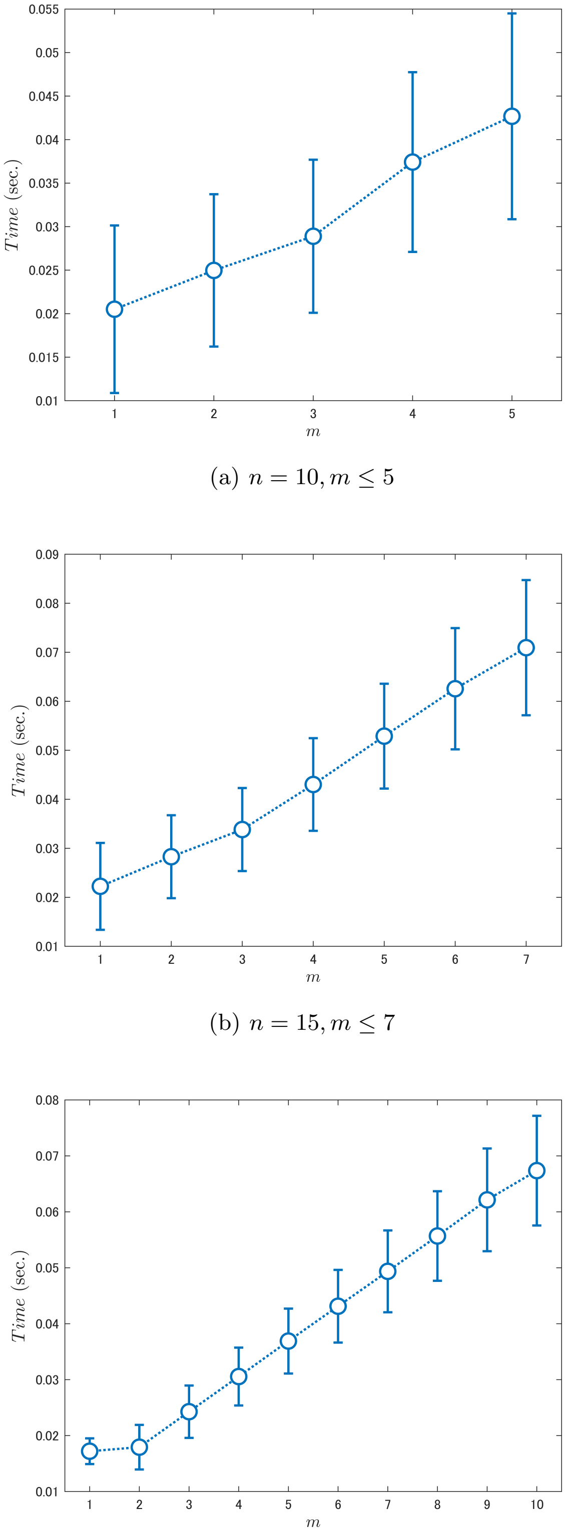
- $n = 10$, and m up to 5, when all combinations are generated from the numbers g in the domain $[0, \binom{10}{m}]$ for $n = 10$, and $m = \{1, 2, 3, 4, 5\}$.
- $n = 15$, and m up to 7, when all combinations are generated from the numbers g in the domain $[0, \binom{15}{m}]$ for $n = 15$, and $m = \{1, 2, 3, 4, 5, 6, 7\}$.
- $n = 20$, and m up to 10, when all combinations are generated from the numbers g in the domain $[0, \binom{20}{m}]$ for $n = 10$, and $m = \{1, 2, 3, ..., 10\}$.

We believe the above experimental instances are useful to evaluate a the generation of combinations within standard computing environments. Thus, the applicability to other contexts becomes straightforward.

To portray the complexity function experimentally, Fig. 4 shows the behaviour in terms of computing time over all unranking instances. By observing the above pictures, we can note the following facts:

- The time to unrank any combination for n up to 20 and m up to 10 is than 80 miliseconds.
- The time to unrank combinations show the linear-like behaviour.

Our results suggest that the use of parallelism is beneficial to improve the efficiency frontiers of our proposed unranking algorithm. We believe our results provide useful and foundational tools to realize the intelligent sampling, evaluation and generation of combinatorial objects with running performance being linear on m and independent of n, which is meritorious in applications when n is large or time-varying.



(a) $n = 10, m \leq 5$

(b) $n = 15, m \leq 7$

(c) $n = 20, m \leq 10$

Fig. 4. Computational efficiency when unranking combinations for various n and m.

4 Concluding Remarks

In this paper, we have proposed an approach for generating combinations from given integer numbers by using evaluations in CPU/GPU, and optimizations over a parallelizable log-concave objective function. The basic idea of our proposed approach is to generate combinations by solving minimization problems of a tailored cost function, which is evaluated by reductions in a CUDA-enabled Graphics Processing Hardware.

Our exhaustive computational experiments involving the generation of combinations over a complete and partial set show the practical efficiency of our proposed approach. In particular, our observations show that the parallelization of the proposed log-concave objective function brings the potential benefits for scalability, and the time to generate combinations show a linear-like behaviour.

In future work we aim at evaluating the generalization of our proposed approach to tackle combinatorial problems involving graphical structures in Decision Sciences, e. g. learning graph topologies of Bayessian Networks and Genetic Programming by number-based representations. We believe our results provide useful and foundational tools to allow representation and sampling of solutions in non-linear combinatorial problems given integer numbers, and its application is straightforward by nature-inspired approaches. Furthermore, our approach offers the generation of m out of n with n-independency, which is meritorious when n is larger compared to m, or n is time-varying.

References

1. Myers, A.F.: k-out-of-n: G system reliability with imperfect fault coverage. IEEE Transactions on Reliability **56** (2007) 464–473
2. Tamada, Y., Imoto, S., Miyano, S.: Parallel algorithm for learning optimal bayesian network structure. Journal of Machine Learning Research **12** (2011) 2437–2459
3. Imada, T., Ota, S., Nagamochi, H., Akutsu, T.: Enumerating stereoisomers of tree structured molecules using dynamic programming. J. Math. Chem. **49** (2010) 910–970
4. Suzuki, K., Yokoo, M.: Secure combinatorial auctions by dynamic programming with polynomial secret sharing. In: Sixth International Conferene on Financial Cryptography, Lecture Notes in Computer Science. Volume 2357. (2003)
5. Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., Verkamo, A.I.: Fast discovery of association rules. Advances in Knowledge Discovery and Data Mining (1996) 307–328
6. Khachiyan, L., Boros, E., Borys, K., Elbassioni, K., Gurvich, V., K., M.: Enumerating spanning and connected subsets in graphs and matroids. J. Oper. Res. Soc. Jpn. **50** (2007) 325–338
7. Amari, Y., Tanaka, U.: A fast algorithm for enumerating bipartite perfect matchings. In: 12th International Symposium on Algorithms and Computation, Lect. Notes Comput. Sci. Volume 2223. (2001) 367–379
8. Chisholm, B.J., Webster, D.C.: The development of coatings using combinatorial/high throughput methods: a review of the current status. Journal of Coatings Technology and Research **4** (2007) 1–12
9. Eiter, T., Makino, K.: On computing all abductive explanations from a propositional horn theory. Journal of the ACM **5** (2007)
10. Scott-Phillips, T.C., A., B.R.: Why is combinatorial communication rare in the natural world, and why is language an exception to this trend? Journal of the Royal Society Interface **10** (2013)
11. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research **15** (2014) 1929–1958
12. Duong, Q., Vorobeychik, Y., Singh, S., Wellman, M.P.: Learning graphical game models. In: Proceedings of the 21st International Jont Conference on Artifical Intelligence. (2009) 116–121
13. Buckles, B.P., Lybanon, M.: Algorithm 515: Generation of a vector from the lexicographical index. ACM Transactions on Mathematical Software **3** (1977) 180–182
14. Crouse, D.F.: Remark on algorithm 515: Generation of a vector from the lexicographical index combinations. ACM Transactions on Mathematical Software **33** (2007) 15
15. Butler, J.T., Sasao, T.: Index to constant weight codeword converter. Reconfigurable Computing: Architectures, Tools and Applications (2011) 193–205
16. Er, M.C.: Lexicographic ordering, ranking and unranking of combinations. International Journal of Computer Mathematics **17** (1985) 277–283
17. Tang, C., Du, M., Lee, R.: Unranking combinations in parallel. In: Proc. Int. Computer Symposium. (1984) 1006–1010

18. Akl, S.: Design and analysis of parallel algorithms. Prentice Hall (1989)
19. Kokosinskiński, Z.: Unranking combinations in parallel. In: Second International Conference Parallel and Distributed Processing Techniques and Applications. (1996) 79–82
20. Kapralski, A.: New methods for generation permutations, combinations and other combinatorial objects in parallel. J. Parallel and Distrib. Computing **17** (1993) 315–326
21. Kokosinskiński, Z.: Algorithms for unranking combinations and other related choice functions. Report 95-1-006, The University of Aizu (1995)
22. Nihenjuis, A., Wilf, H.S.: Combinatorial Algorithms for Computers and Calculators. Academic Press, New York (1978)
23. Kreher, D., Stinson, D.: Combinatorial Algorithms: Generation Enumeration and Search. CRC Press (1998)
24. Nayak, A., Stojmenovic, I.: Handbook of Applied Algorithms: Solving Scientific, Engineering and Practical Problems. Wiley-Blackwell (2008)
25. Savage, C.: A survey of combinatorial gray codes. SIAM Review **39** (1997) 605–629
26. Liu, C.N., Tang, D.T.: Distance-2 cyclic chaining of constant-weight codes. IEEE Transactions on Computers **C-22** (1973) 176–180
27. Shimizu, T., Fukunaga, T., Nagamochi, H.: Unranking of small combinations from large sets. Journal of Discrete Algorithms **29** (2014) 8–20
28. Woodward, J.R., Bai, R.: Canonical representation genetic programming. In: Proceedings of the First ACM/SIGEVO Summit on Genetic and Evolutionary Computation. GEC '09, New York, NY, USA, ACM (2009) 585–592

Genetic Algorithms for Optimizing Nanostores' Routing in Emerging Markets

A. Sabiri[1] and F. Riane[2]

1. Hassan First University, Laboratoire d'Ingénierie, Mécanique, Management Industriel et Innovation, Settat, Morocco
Sabiri.asmaa@gmail.com

2. Ecole Centrale Casablanca, Laboratoire de Génie Industriel Centrale Supélec Paris, Morocco
fouad.riane@centrale-casablanca.ma

Keywords : High Frequency Stores, Assignment Problem, Multiple Depot Routing, Traveling Salesman, Emerging Countries, Urban logistics, Metaheuristics.

1 Introduction

In an escalating need to find new markets to grow and expand, more companies are heading toward emerging markets. Substantial opportunities of growth are sought there as well as important challenges to face. Companies that benefit the most are those of consumer-packaged goods (CPG), fast-moving consumers goods (FMCG), pharmaceuticals and medical devices as well as consumer electronics and high-tech equipment. Emerging markets are characterized by young economies and growing populations that are driving new cultures of consumption. Trade transactions are of low value but high volumes. Modern and traditional trade channels coexist in the same area with the dominance of the latter.

Traditional trading consists of largely of informal channels, such as open-air markets, individual vendors, including street vendors, tabletop merchants etc. It also includes thousands of little structured outlets, family-operated known as nanostores or High Frequency Stores. These small retailers are the main source of CPG and FMCG for many consumers in Africa, Asia and Latin America. They offer affordability, convenience, the right mix of items and can easily gain the trust of their customers [1].

On the other hand, they are not really sophisticated, and companies may experience gaps between supply and demand very often. Thus, they must correctly design their distribution network and well manage their points of sales coverage in each territory. They face challenges from network design to urban logistics through assortment and delivery planning. The problem of concern in this research deals with point-of-sales coverage design and optimization. Sales coverage models enable companies to increase revenue with existing customers and acquire new customers through an effective application of sales and marketing resources.

2 Problem statement

To address the challenges above, one needs to develop models and methods for business insights that facilitate efficient salespersons' assignments while covering salespoints efficiently. These models connect with two main research fields: Territory Design field and Traveling Salesman field. The aim of a territory design or districting problem is partitioning a large territory into districts or areas. It can be applied in political districting, sales territory design, school districting, etc. Districts are usually designed by optimizing three main features: compactness of districts, dissimilarity of partitions between different periods of time and profit equity between salesmen [2]. The Traveling Salesman problem then intervenes because we need to optimize routs (travelling cost) travelled by salesmen alongside districting.

Our problem can be stated as follow. Given a number of salespersons (SP), each of which has to visit a predefined set of salespoints (nanostores). Depending on its importance, a nanostore may be visited once or several times over a planning horizon (which is a week in our setting). Every day, SPs start by a morning meeting at a gathering point for a daily commercial brief to set everyone's actions and to bring back the commercial objectives of their activity. SPs start their tours with a set of pre-assigned sale points to visit according to a predefined planning. The gathering point (GP) should be appropriately located since the SPs must return back to it at the end of each day to debrief on their tours performances.

A salesperson works 7 hours a day. Each stop lasts a certain amount of time, depending on the size of the sale point. During this time, the SP has many tasks, including general relationship management, merchandising, selling in new initiatives, order taking, capturing sales fundamentals and competitive intelligence data. Workload must be well balanced between SPs. A salesperson is accountable for his sales' performance. Therefore, nanostores must be fairly assigned with respect to their profit potential. Also, sale points whom have been already allocated to some SPs in the past, would stay with the same SPs rather than to be approached by new ones. SPs ususllay keep a close relationship with the retailers. The objective behind solving the problem is to maximize the global profit by assigning nanostores to sales representatives. The targeted assignments have to minimize pre-assignments' disturbance and the variability of profit potential between SPs, while respecting the workload of each one of them and the visiting frequency of each store.

The problem depicted above can be modeled as a multi-depot multiple traveling salesman problem (mTSP) with a constraint of time window. Defining briefing points and optimizing tours at the same time need a dynamic solving approach. We assume that there's one briefing point. Its location is known and remains fixed over the planning horizon. Let us denotes the distance between this gathering point (GP) and the nanostore (i). The TSP is usually formulated using an assignment-based double-index integer linear programming formulation [3]. We will be using a similar formulation to our problem. We consider a two-echelon supply chain in which products are to be pushed from a central common warehouse to a set of nanostores over a time horizon H.

We consider $SP = \{salesperson\ j, 1 \leq j \leq k\}$ the set of salespersons.
$n\}$ is the set of nanostores to be visited during a planning horizon including m new ones that have never been assigned before. $H = \{Horizon\ t, 1 \leq t \leq T\}$ defines the considered time horizon. A salesperson is assigned a working territory consisting on a list of stores to be visited over the horizon H according to a pre-established schedule. X_{ijt} is be the binary variable that determines if a store (i) is assigned to a salesperson (j) at period (t). Each store is attributed a visit frequency f_i depending on its profile and sales' volume. Based on the experience of each salesperson regarding its previous missions, the company computes an index for each store to capture its previous assignments to every salesperson. The main elements of our model identify as follows:

Profit potential balance constraints (1): Salespersons performance and thus reward are evaluated based on their profit contribution. Therefore, for them to have the same reward chance, the profit potential P_i of the assigned stores has to be equivalent for each salesperson. The profit potential balance can be stated by inequations **(1).** The difference between the profit gained by a salesperson *h* and the average profit gained over the horizon *H* by all the *K* sales representatives, should not exceed a variability limit *VAR1*.

$$\begin{cases} \quad - \qquad - \qquad - \\ \quad - \qquad - \qquad - \end{cases} \qquad \textbf{(1)}$$

The workload constraints (2) combine the travel time from the starting point to each assigned store with the stopping time at each store T_i. This total combined time could not exceed the maximum workload associated with a salesperson *j* during a period *t* denoted by parameter WL_{jt}. In the literature, the tour cost of districts is very often approximated by the Beardwood–Halton–Hammersley theorem [4], when having a multiple objective problem alongside a traveling salesman problem, but it is nonlinear, so it can't be used in linear resolution. We assume in a first approximation that the total distance travelled by a sales representative can be estimated using the radial distance as the sum of distances D_i between the gathering point (GP) and each visited store *(i)*. *V* is the average traveling speed of the salespersons.

$$- \qquad \textbf{(2)}$$

Workload balance constraint (3): Guaranteeing equivalent chances for salespersons comes down as well to balance their workload. Constraint **(3)** minimizes the disparity between the salespersons working hours through a period.

(3)

Visits' number constraint (4): At every period *t*, each store *i* should be visited by only one salesperson. It translates to the formulation in **(4)**.

(4)

Visiting frequency constraint (5): For a store to attain its maximum profit, it should be visited at least f_i times, as formulated in **(5)**. Whereas new stores may be visited lesser times because their frequency is only estimated, and the workload may not be sufficient to visiting all of them at once.

(5)

Disturbance constraint (6): We want to minimize pre-assignments' disruption, as this latter is directly correlated with profit generation. Disruption is undesirable to capitalize on the established relationships between salespersons and retailers. Let be the disturbance index that compares assignments of period *t* with period *t-1*. and for each store NS_i. We can therefor express our constraint as in **(6)**, where DIS is the maximum disturbance allowed. This constraint can also be formulated as a penalty constraint that increases every time a NS is visited by more than one SP during H.

(6)

As for the objective function, we tried out 3 combinations of profit balance and workload balance constraints that are expressed in the formulation **(7)**, **(8)** and **(9)**. (7) is with workload balance constraints (3), (8) with workload balance constraints (1) and (9) a combination of the two. To linearize the absolute value in $\boldsymbol{f_1}$, we minimize $\sum_{j=1}^{k} \boldsymbol{H_j}$ subject to $\boldsymbol{H_j} \geq \boldsymbol{PV_j}$ and $\boldsymbol{H_j} \geq -\boldsymbol{PV_j}$. The same applies for $\boldsymbol{f_2}$ and $\boldsymbol{f_3}$.

(7)

(8)

(9)

3 Resolution

In a first linear resolution, we used real data of an urban area territory where a mailing company operates. We dismissed the disturbance constraint for the tests because of non-linear complexity. For future modelling, and to solve the model including the disturbance constraint and the tour cost minimization, we will be using genetic algorithms to cope with the problem complexity.

Tours cost minimization, workload equity, profit balance and disturbance should all be optimized to give a maximum profit in a reasonable time. The definition of the gathering points (GPs) is also crucial. Their number and their location should be optimized dynamically with the problem resolution. It can be also evaluated at the end of each instance and relocated.

For the genetic algorithm model, we suggest using 2-dimentional chromosome for all salesmen related to all depots. Rows represent salespoints and columns represent days. Each number represent a salesman. "0" means a salespoint will not be visited in a specific day. A salesman can be assigned to multiple rows each day. The constraints of cost, profit balance, and disruption can be formulated as a weighted sum of 3 different fitness functions.

4 References

[1] E.E. Blanco, J.C. Fransoo (2017). Reaching 50 Million Nanostores: Retail Distribution in Emerging Megacities. *CreateSpace Independent Publishing Platform*

[2] B. Bozkaya, E. Erkut, D. Haight, G. Laporte (2011). *Designing new electoral districts for the city of Edmonton*, Interfaces vol. 41(6), pp. 534-547. Maryland USA

[3] D. Davendra (2010). Traveling Salesman Problem, Theory and Applications, *Intech*, Croatia

[4] J. Beardwood, J.H. Halton, J.M. Hammersley (1958). *The shortest path through many points, Mathematical proceedings of the Cambridge Philosophical Society*, vol. 55, no.4, pp.299–327, Cambridge University Press

A cooperative multi-swarm particle swarm optimizer based hidden Markov model

Oussama Aoun[1] and Abdellatif El Afia

1.ENSIAS - Institute of computer science, Mohamed V University, Rabat, Morocco.
oussama.aoun@um5s.net.ma
a.elafia@um5s.net.ma

Abstract Particle swarm optimization (PSO) is a population-based stochastic metaheuristic algorithm; it has been successful in dealing with a multitude of optimization problems. Many PSO variants have been created to boost its optimization capabilities, in particular, to cope with more complex problems. In this paper, we present a new multi-population particle swarm optimization with a cooperation strategy. The proposed algorithm splits the PSO population into four sub swarms and attributes a main role to each one. A machine learning technique is designed as an individual level to allow each particle to determine its suitable swarm membership at each iteration. In a collective level, cooperative rules are designed between swarms to ensure more diversity and realize the better solution. Several simulations are performed on a set of benchmark functions to examine the performances of this approach compared to a multitude of state of the art of PSO variants. Experiments reveal a good computational efficiency of the presented method with distinguishable performances.

Keywords: particle swarm optimization, multi-swarm, population control, hidden markov model.

1 Introduction

Nowadays, a number of cooperative approaches have been emerged to improve swarm intelligence optimizers. In particular, various efforts have been made to improve the collective search behavior of optimization algorithms mainly in swarm intelligence approaches such as particle swarm optimization [13] and ant colony optimization [9]. The main purpose of these essays is to incite complex global behaviors through local interactions by sharing information between different agents and improving learning capacity. Furthermore, it may help them to adapt to unexpected variations (dynamic optimization) when they are communicating with other agents.

Concerning particle swarm optimization (PSO), one of its challenging issues is to formalize the design of such cooperative multi-swarm behavior of agents (which are named particles in the case of PSO). It main advantage is to enhance the diversity of the algorithm or to achieve a trade-off between exploration and exploitation. A commonly used cooperation form of PSO is based on the idea of considering multi-swarms (multi-populations), it consists in dividing the whole search space into local subspaces, each of which might cover one or a small number of local optima, and then separately searches within these subspaces. Another way to define cooperative PSO is to assign different roles to particles. Thus, different particles can play different roles, and each one of these particles can play different roles during the search processes. A challenging task within this PSO variant is how each particle has to decide which role it will assume. In this paper, the machine learning algorithm hidden Markov model (HMM) is applied in an individual level (each particle) to model how the decision making of particles to choose the adequate sub-swarm which it will belong. That is, HMM is used to learn and predict the most probable swarm, corresponding to each particle in order to control particles behavior of PSO. This process is performed through the Viterbi algorithm that gives the most likely path of states for each particle at each PSO iteration. Or, for each sub swarm, an associated role is given: exploration, exploitation, convergence and jumping out. Then, in a collective level of the swarm, a cooperative design is made to guide the search and move toward different promising sub-regions. A Master/Slave scheme is chosen for setting cooperation rules between sub-swarms.

The rest of the paper is organized as follows: in the next section, we outline the related works. In section 3, we present our approach. Section 4 presents the obtained results for the experiments. Finally, we conclude and present perspectives to our work.

2 Literature review.

In recent years, there has been increased interest in the use of learning methods inside PSO to control its behavior and then to improve its performance. That is, various methods have been proposed to control PSO behavior and to improve the particles learning ability. We can differentiate between two kinds of control approaches which have been used inside PSO in the literature. In the first one, the control depends on the iteration and then the whole swarm follow the same strategy (as it is done in our previous work [1]). In the second one, the control depends on the particle itself. That is, at each iteration, particles are grouped into sub-swarms, and the particles of each swarm have a specific role in the swarm. This second case is our focus in this paper and then we present in the following a review of papers which interested in this issue.

Firstly, [14]proposed four operators which play similar roles as the four of states the adaptive PSO defined in [29], which are exploration, exploitation, convergence and jumping-out. Their approach is based on the idea of assigning to each particle one among different operators based on their rewards. Moreover, [30] defined a cooperative approach to PSO based on dividing the population on four sub-swarms according to the four states. Furthermore, this type of control is related to the concept of cooperative swarms which has been introduced by [25]. This principle has been achieved in their paper by using multiple swarms to optimize different components of the solution vector cooperatively. We can see that this variant of PSO (multi-swarm PSO) has been presented in the literature as a specific and separate algorithm known by multi-swarm optimization [5]. In the proposed variant, the authors have been inspired by the quantum model of atoms to define the quantum swarm. Also, another grouping approach has been suggested by [19]. Another cooperative PSO approach can be defined by clustering approaches as proposed by [28]. Their approach consists of assigning particles to different promising sub-regions basing on a hierarchical clustering method.

More generally, four main categories have been proposed to improve PSO performance, which are: configuration of the parameters (adaptive control), the study of neighborhood topology of particles in the swarm of PSO, hybridization with other optimization algorithms and integration of learning strategies (diversity control). Concerning the two types mentioned at the beginning of this section. The former correspond to the first type, while the latter is related to the second one. Furthermore, the control the PSO parameters has been proposed in a number of papers with the purpose of achieving a trade-off between the diversity and the convergence speed. It has generally been done using learning strategies such as the comprehensive learning [16] approach in where each particle learns from another particle which is chosen according to a learning probability. Concerning hybridization, it is a long-standing of PSO and example of improvement can be found in [23].

We can see from the literature that many papers have inspired from some approaches used in multi-agent systems to defined automated cooperative approach. An example of using the multi-agent concept in PSO can be found in [7]. That is, incremental social learning which is often used to improve the scalability of systems composed of multiple learning agents has been used to improve the performance of PSO. Furthermore, [2] proposed a multi-agent approach which combines simulated annealing (SA) and PSO, we can remark that their idea is related to the generic notion of hyper-heuristics which consists of finding the most suitable configuration of heuristic algorithms. [8] has cited the may features obtained by using agents in configuring metaheuristics, which are distributed execution, remote execution, cooperation and autonomy.

This issue (the interaction between swarm intelligence and multi-agent systems) has been given much attention in the last few years in particular by the popularization of the swarm robotic field. In particular, [4] affirmed the concept of swarm appears nowadays closely associated with intelligent systems in order to carry out useful tasks. The author also analyzed qualitatively the impact of automation concepts to define the intelligent swarms. Moreover, [6] have outlined the main characteristics of swarm robotics and analyzed the collective behavior of individuals in some fields. They affirmed that finite state machines are one of the most used adequate approaches to model this behavior. Another commonly used approach for this purpose is reinforcement learning.

In particular, the using of multi-agent concepts can be useful to self-organize particles in PSO using simple rules as defined by [3].Their main idea was to define six states, which are cohesion, alignment, separation, seeking, clearance, and avoidance. Furthermore, the finite state machine has been used for movement control. That is, the states have been defined by a collection of velocity components and their behavior specific parameters. Furthermore, the population has been divided into two swarms in order to introduce the divide and conquer concept using genetic operators. Another automation approach which can be used inside PSO is cellular automata (CA). It can be used for instance split the population of particles into different groups across cells of cellular automata. [24] has integrated it in the velocity update to modify the trajectories of particles.

In term of multi-swarm design of PSO, [15] provided a multi-swarm and multi-best for the particle swarm optimization algorithm. They randomly split particles into multi populations. This algorithm updates velocities and positions of particles using multi-gbest and multi-pbest rather than single gbest and pbest. [18] proposed a novel variant known as Center PSO, it makes use of an extra particle as a center particle that controls the search direction of the entire swarm. Also, [20] built a Multi-swarm cooperative particle swarm optimizer based on a master-slave model; the slave swarms perform as a single PSO while the master swarm iterates depending on its knowledge as well the knowledge of the slave swarms.

At the light of the literature in term of enhancing PSO performances and building a more efficient variant of this algorithm, this paper addresses a new variant of PSO based on cooperative multi-swarm design with a coefficient adaptation.

3 Cooperative multi-swarm conception of PSO

In this section, the standard PSO algorithm is defined with its parameters. Then, the way how sub-swarms are identified is given depending on the individual particle state given by HMM. Each sub-swarm will its own configuration of the parameters of its particles. Cooperation rules will be defined to ensure the information exchange between subs warms during the search process.

3.1 The standard PSO

The standard PSO is a population based metaheuristic algorithm introduced firstly by [13]. Its mechanism starts with a population of random solutions and during a search process, particles are looking for optima by moving in the search space. In PSO, particles are potential solutions in a D-dimension search space, having a velocity that is adjusted dynamically depending on both individual and social experience. Therefore the velocity and the position of every single particle is updated based on to Eqs. (1) and (2)

$$v_i(t+1) = w\, v_i(t) + c_1 r_1 (pBest - x_i(t)) + c_2 r_2 (gBest - x_i(t)) \quad (1)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (2)$$

Where r_1 and r_2 are two are two vectors of D dimention provided by distribution function of independent uniform random numbers defined between 0 and 1. pBest is the best position of the particle and gBest is the global best of the entire swarm. In our case of multi swarms, a multi gBest positions are defined. w is the inertia weight, and c_1 and c_2 are the acceleration coefficients. Equation (1) is involved to compute the particle new velocity, when Equation (2) is used to update the position of the particle using its previous position and its new velocity.

More detail of these parameters can be seen for example in [21].

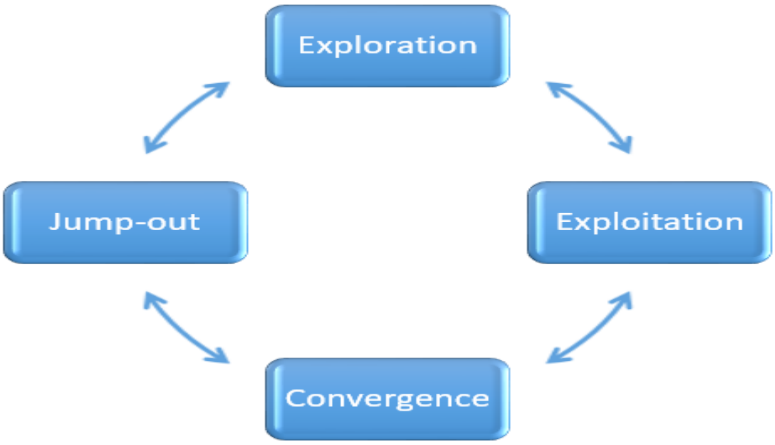
In our approach, we interest especially on dividing the population into multi-population or also called multi-swarm. Each swarm will has its own characteristics and search behavior. The next paragraph will gives how sub swarms are constituted and how each sub swarm will be customized to a specific role in the search space.

3.2 Sub-swarms constitution

In this approach swarm is divided to a sub-swarms in the objective to achieve a good trade-off between the population diversity and the convergence speed, and specially a good management of the exploration and exploitation of the search process during execution in order to attain the best possible solution in the minimum number of iterations. Inspired from the definition of [29] of the evolutionary states for PSO, each sub swarm will groups particles of a specific four evolutionary state that are: Exploration, exploitation, convergence and jumping-out.

Then, each particle is viewed as a Markov chain having a state $\{s_i\}_{i\in[1,4]}$. During iterations, a particle can has a specific state i that design its membership to a specific swarm i. Also, a particle can change the state from iteration to another and change consequently its corresponding sub swarm. So, a movement between sub swarms is indicated by the rows in the figure 1.

Figure 1 Sub-swarms and possible particle movements



To model the associated swarm of particle, an associated markov chain with state $\{s_i\}_{i\in[1,4]}$ is defined to each particle. However, particle state cannot be perceived directly but only by observing some key parameters across iteration. Hence, a hidden markov chain is defined for each particle as a by a triple $= (\Pi, A, B)$, all processes are defined on a probability space (Ω, F, P) :

- $\Pi = (\pi_i)$ The vector of the initial probability distribution over states;
- A=(a_{ij}) The state transition matrix, $P(q_t = i|q_{t-1} = j),\ i, j \in [1, N]$
- $B = (b_{jk})$ The emission matrix also called the confusion matrix, $P(o_t = k|q_t = j), j \in [0, N], k \in [0, M]$.

The set of N states $\{q_t\}_{t\in N}$ takes values from the set $S = \{s_i\}_{i\in[1,4]}$ what references respectively: exploration, exploitation, convergence, and jumping out. The change of state is reflected by the PSO sequence $(q_1 = s_2)\Rightarrow(q_2 = s_1)\Rightarrow(q_3 = s_2)\Rightarrow\ldots$, as deduced by [29], corresponding to the Markov Chain.

Furthermore, we define corresponding initial transition probabilities, $P(q_t = i|q_{t-1} = j),\ i, j \in [1,4]$. This probability controls all behavior of transition between states of APSO resolution. We take for all possible i and j transitions a transition probability of 0.5.

The initial state probability corresponds to deterministic start in exploration state:

$$\Pi = (\pi_i) = [1\ 0\ 0\ 0] \qquad (3)$$

The observed parameter of this hidden chain is the evolutionary factor f (defined in [29]) of the APSO.

Observations will be belonging f to subintervals of [0,1] ([0,0.2], [0.2,0.3], [0.3,0.4], [0.4,0.6], [0.6,0.7], [0.7,0.8], [0.8,1]). We divide [0,1] to seven subintervals, so the set observation $Y = \{y_i\}_{i \in [1,..,7]}$ will be number of the subinterval witch belong f. Let $sub: [0,1] \rightarrow \{1,2,..,7\}$ the function that returns the corresponding interval of f, it corresponds also to the observation.

$$sub(f) = \delta_{[0,0.2[}(f) + 2\delta_{[0.2,0.3[}(f) + 3\delta_{[0.3,0.4[}(f) + 4\delta_{[0.4,0.6[}(f) + 5\delta_{[0.6,0.7[}(f) + 6\delta_{[0.7,0.8[}(f) + 7\delta_{[0.8,1]}(f) \qquad (7)$$

$$\left(with \quad \delta_{[a,b]}(x) = \begin{cases} 1, x \in [a,b] \\ 0 \; otherwise \end{cases} \qquad a, b \in \mathbb{N}, x \in \mathbb{R}\right)$$

Emission probabilities are deduced from defuzzification process (Figure 1) of [29] as follow, the same as in [1]. :

$$P = \begin{bmatrix} 0 & 0 & 0 & 0.5 & 0.25 & 0.25 & 0 \\ 0 & 0.25 & 0.25 & 0.5 & 0 & 0 & 0 \\ 2/3 & 1/3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/3 & 2/3 \end{bmatrix}$$

After initializing HMM parameter, Baum-welch algorithm (algorithm 3) is used at each iteration to estimate and update HMM emission and transition matrix; this allows HMM to be more adaptive and accurate in the classification step.

Then, the Viterbi Algorithm is used with the estimated parameters to find the most probable sequence associated with hidden states given a sequence of observed states. The algorithm will find the max Q (state sequence $Q = q_1 q_2 \dots . q_T$) for a given observation sequence ($O = o_1 o_2 \dots . o_T$) by means of induction (t the iteration number). It is about to find the highest probability paths for states [22]. Viterbi algorithm determine then how each particle move between sub swarms.

Indeed, HMM has the ability to learn states of our automata from hidden observation based on the maximum likelihood estimation[10], this learning feature of HMM is used to control the particles cross PSO iterations.

3.3 Sub-swarms parameters adaptation

Moreover, according to each sub swarm associated to each state, PSO parameters are adjusted, especially acceleration parameters c_1, c_2 and inertia weight w with elastic learning in convergence sub-swarm [29]. It is done based on APSO parameters update in [29] and [1], see algorithm 1 :

```
Algorithm 1: Adaptive acceleration update in APSO [29]
Data: Position and accelerations factors
Initialization: positions and accelerations factors c1 and c2 ;
if sub-swarm = exploration then Increasing c1 and Decreasing c2 ;
else if sub-swarm = exploitation then
        Increasing c1 and Slightly Decreasing c2
else if sub-swarm = jumping out then
        Increasing Slightly c1 and Increasing c2
end
else if sub-swarm = convergence then
        Decreasing c1 and Increasing c2
end
Return c1 and c2
Result: Updated acceleration factors
```

For all sub swarms, the inertia weight is set as follow:

$$\omega(l) = \frac{1}{1 + 1.5e^{-2.6l}} \in [0.4, 0.9] \; \forall \, l \in [0,1]$$

A real-time state estimation procedure is performed to identify each particle adequate swarm: exploration, exploitation, convergence, and jumping out. It qualifies an automatic control of the sub-swarms.

3.3 multi-swarms cooperation:

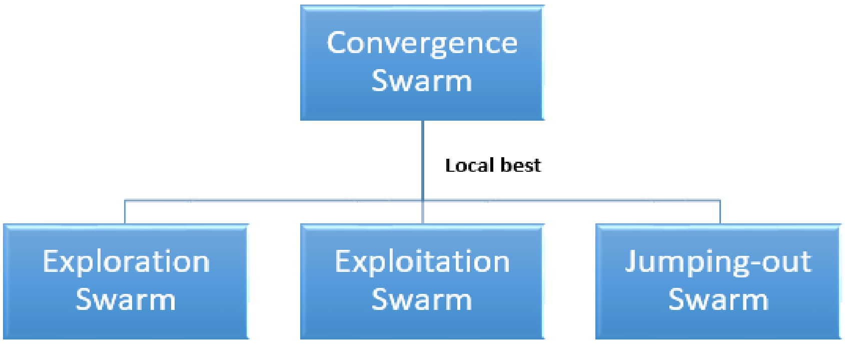
To make use of the multi-swarm design given in the previous paragraphs, it's mandatory to set a cooperation model to make use of the search capabilities given by each sub-swarm. A master/slave cooperation model is chosen in this approach like in [<niu2007mcpso>], where the slave swarms perform as a single PSO while the master swarm iterates depending on its knowledge as well the knowledge of the slave swarms. In our case, the master swarm is the swarm associated to the convergence state. Then, the slave swarms will be those associated to exploration, exploitation and jumping-out states.

Each slave swarm with some n particles adapts itself according to its own evolutionary attached state separately. So, a slave swarm can be viewed as an independent swarm not connected to the other slaves. For the master swarm, the particles improve themselves not simply depending on the social knowledge of the master swarm but as well as that of the slave swarms. This notion is made by additional integrating a new dimension on the velocity of the particles in its velocity update. The equations for the velocity update of the master swarm will be:

$$v_i^C(t+1) = w\, v_i^C(t) + c_1 r_1 \left(pBest - x_i^C(t)\right) + c_2 r_2 \left(gBest^C - x_i^C(t)\right) + c_3 r_3 (gBest^s - x_i^C(t)) \quad (4)$$

Where C represents the convergence sub-swarm, $c_3$is called migration coefficient, r_3 uniform random sequence in the range [0, 1], $gBest^C$ is the is the global best of the convergence swarm and $gBest^s$ is the is the global best of the other slave sub swarms, in particular exploration, exploitation and jumping-out.

Figure 2 Sub-swarms and the master/slave interactions



The figure 2 represents a communication scheme between sub-swarms. Then, the global algorithm of this approach is described in algorithm 2.

Algorithm 2: MultiS-PSO

```
Data: The objective function (f)
Initialization: positions, velocities of particles, accelerations factors of all four swarms; Set t value to 0;
while (number of iterations $t \leq t_{max}$ not met) do
for i = 1 to the number of particles do
    Decoding specific particle state (viterbi ) ;
    Associate particle i to its decoded sub-swarm;
    Update w according to Equation (5) ;
    Update c1 and c2 values according to the corresponding state (algorithm 1) ;
    if convergence swarm
        Update velocities according to Equation (4)
    else    Update velocities according to Equation (2)
    end
    Update positions according to Equation (1)
    compute $f(x_i)$ ;
    For each sub-swarm i:
        if ( $f(x_i) \leq fbest$ ) then
            $fbest \rightarrow f(x_i)$ ;
            $pbest \rightarrow x$ ;
        end
        if ( $f(pbest) \leq fGbest$ ) then
            $fGbest \longrightarrow fbest$ ;
            $gbest \longrightarrow Xbest$ ;
        end
        if sub-swarm = convergence then
            Elistic learning [29];
        End
    end
end
$t \rightarrow t + 1$ ;
end
Return $p_{best}$ and $f_{best}$ ;
Result: The solution based on the best particle in the population and corresponding fitness Value
```

4 Experimentations

In this section, we conduct an experimental analysis fo the proposed conception method of cooperative multi-swarm PSO called MultiS-PSO. Simulations are done on various benchmark functions: unimodal and multi-modal. Then, results are compared with other state of the art of PSO related variants.

4.1 Parameters setting

Ten benchmark functions constitute fitness function used for experimentation which are split to modal and multimodal as shown in Table 1. Cross executions. For every function, the best and the average value are put to use in comparison.

Benchmark functions	Name	Type
$\mathbf{f_1} = \sum_{i=1}^{D}[(10^6)^{i-1/_{D-1}} x_i^2]$	Elliptic	Unimodal
$\mathbf{f_2} = \sum_{i=1}^{D}(\mid x_i + 0.5\mid)^2$	Step	Unimodal
$\mathbf{f_3} = \sum_{i=1}^{D} x_i{}^2$	Sphere	Unimodal
$\mathbf{f_4} = 10^6\, x_1{}^2 + \sum_{i=2}^{D} x_i{}^2$	Tablet	Unimodal
$\mathbf{f_5} = \sum_{i=1}^{D}(\sum_{i=1}^{D} x_i\ )^2$	Quadric	Unimodal
$\mathbf{f_6} = \sum_{i=1}^{D}[\, x_i{}^2 - 10\cos(2\,\pi\, x_i) + 10]$	Rastrigrin	Multimodal
$\mathbf{f_7} = -20\exp(-0.2\sqrt{\frac{1}{D}x_i{}^2})$	Ackley	Multimodal
$\mathbf{f_8} = \frac{1}{4000}\sum_{i=1}^{D} x_i{}^2\ - \Pi\cos(x_i / \sqrt{i}) + 1$	Griewang	Multimodal
$\mathbf{f_9} = \sum_{i=1}^{D} x_i \sin(\sqrt{x_i})$	Schwefel	Multimodal
$\mathbf{f_{10}} = -\frac{1 + \cos(12\sqrt{x_1{}^2} + x_2{}^2)}{1/2\,(x_1{}^2 + x_2{}^2) + 2}$	Drop wave	Multimodal

Table 1 Description of Benchmark functions

Various versions of the PSO algorithm from the literature are selected in the experimentation level for comparison (see table 2). Then, simulations and validations of the proposed on benchmark functions. Parameters are the same $c_1 = c_2 = 2, \omega = 0.9$ for all PSO variants and $c_3 = 0.8$.

The Swarm size is 30 with dimension of 30. Each run contains 1000 generation of optimization process.

Algorithm	Name	Reference
YSPSO	PSO with compressibility factor	[17]
SELPSO	Natural selection based PSO	[11]
SecVibratPSO	Order oscillating PSO	[12]
SecPSO	Swarm-core evolutionary PSO	[26]
SAPSO	Self-adaptive PSO	[11]
RandWPSO	Random inertia weight PSO	[27]
LinWPSO	Linear decreasing weights PSO	[27]
CLSPSO	Cooperative line search PSO	[26]
AsyLnCPSO	Asynchrous PSO	[11]
SimuAPSO	PSO with Simulated Annealing	[26]

Table 2 Compared variants of PSO

4.2 Performance evaluation

Firstly, we dress obtained results of all executions to compare the solution accuracy of our MultiS-PSO. The best and the average values resulted from experimentations are given in table 3:

Functions		APSO	PSO	SimuA-PSO	Sec-PSO	RandW PSO	YSPSO	SelPSO	SecVibr atPSO	SAPSO	LinWPSO	AsyLnC PSO	MultiS-PSO
f_1	Best	49	13566	115719	4689	9843	1420	16382	275	5095	7067	3765	**6.03**
	Mean	150	24618	288102	10930	33384	2726	27998	32132	14850	20280	11045	**38**
f_2	Best	0	5.1e-09	5.8e-06	1.9e-31	2.1e-12	0	8.6e-10	7.3e-10	0	0	0	**0**
	Mean	0	6.5e-05	0.04	9.8e-12	3.6e-05	0	1.8e-05	0.03	0	0	3.1e-30	**0**
f_3	Best	0.01	26.55	93.19	20.58	37.7	6.77	36.01	0.8	18.20	17.66	22.89	**0.0041**
	Mean	0.05	50.15	188.56	38.79	64.09	13.77	63.59	71.05	31.96	40.53	34.58	**0.0068**
f_4	Best	0.02	94.97	251.36	71.4	110.48	23.72	30.47	17.42	37.21	51.24	21.73	**0.0078**
	Mean	0.05	135.82	396.21	110.48	246.78	42.31	77.51	199.04	84.79	95.74	44.32	**0.0133**
f_5	Best	16435	629e+5	587e+6	421e+5	102e+6	127e+4	839e+5	107e+6	459e+5	165e+5	131e+5	**2697**
	Mean	67851	196e+6	210e+7	978e+5	434e+6	843e+4	210e+6	562e+6	166e+6	155e+6	384e+5	**6684**
f_6	Best	8.24	266.20	358.22	208.54	293.88	142.44	285.16	221.93	165.66	170.64	193.83	**6.89**
	Mean	16.14	307.24	462.02	262.89	322.35	175.84	315.60	344.76	273.31	261.77	291.54	**14.70**
f_7	Best	0.03	4.79	6.9436	4.8963	5.403	3.1785	5.665	1.2753	3.8279	4.7261	5.8372	**0.001**
	Mean	0.31	5.61	8.6336	5.2716	6.5613	4.2219	6.1951	4.4528	5.2531	5.6829	6.8189	**0.016**
f_8	Best	7.50e-5	0.17	0.52	0.15	0.25	0.05	0.27	0.05	0.13	0.14	0.08	**1.5e-4**
	Mean	0.01	0.39	0.87	0.25	0.45	0.12	0.41	0.42	0.25	0.31	0.23	**2.1e-4**
f_9	Best	-118.35	-3e+287	-7.2+47	-4+158	-	-3+34	-1e+308	-	-1+231	-1e+220	-3e+47	**-118.35**
	Mean	-118.34	-3e+286	-1e+47	-9e+157	-	-3e+33	-	-	-1e+230	-1e+219	-3e+46	**-118.34**
f_{10}	Best	-1	-1	-0.92	-1	-0.94	-1	-0.99	-0.93	-1	-1	-1	**-1**
	Mean	-1	-0.95	-0.74	-0.96	-0.93	-0.98	-0.95	-0.82	-0.97	-0.96	-0.98	**-1**

Table 3 Results comparisons with other variants of PSO

Our proposed method provides the much better results than all state of the art PSO variants used for comparison. MultiS-PSO has enhanced a significant better performances in term of solution accuracy for both unimodal and multimodal functions.

In term of convergence speed, comparisons are illustrated in figure 3.

.

Figure 3 Comparison on convergence speed

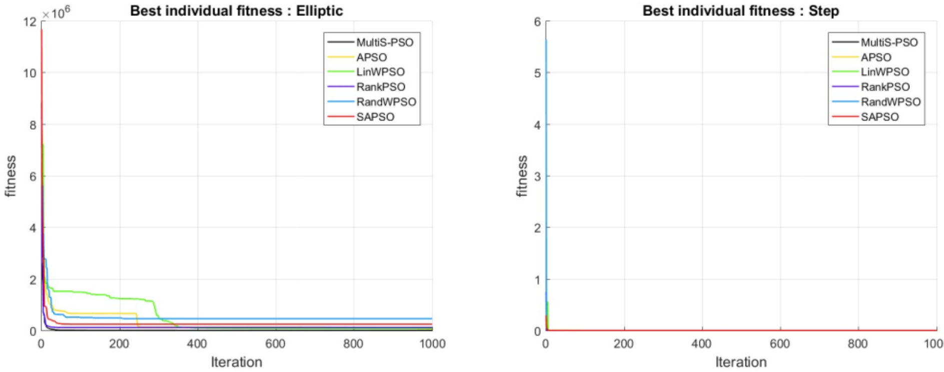
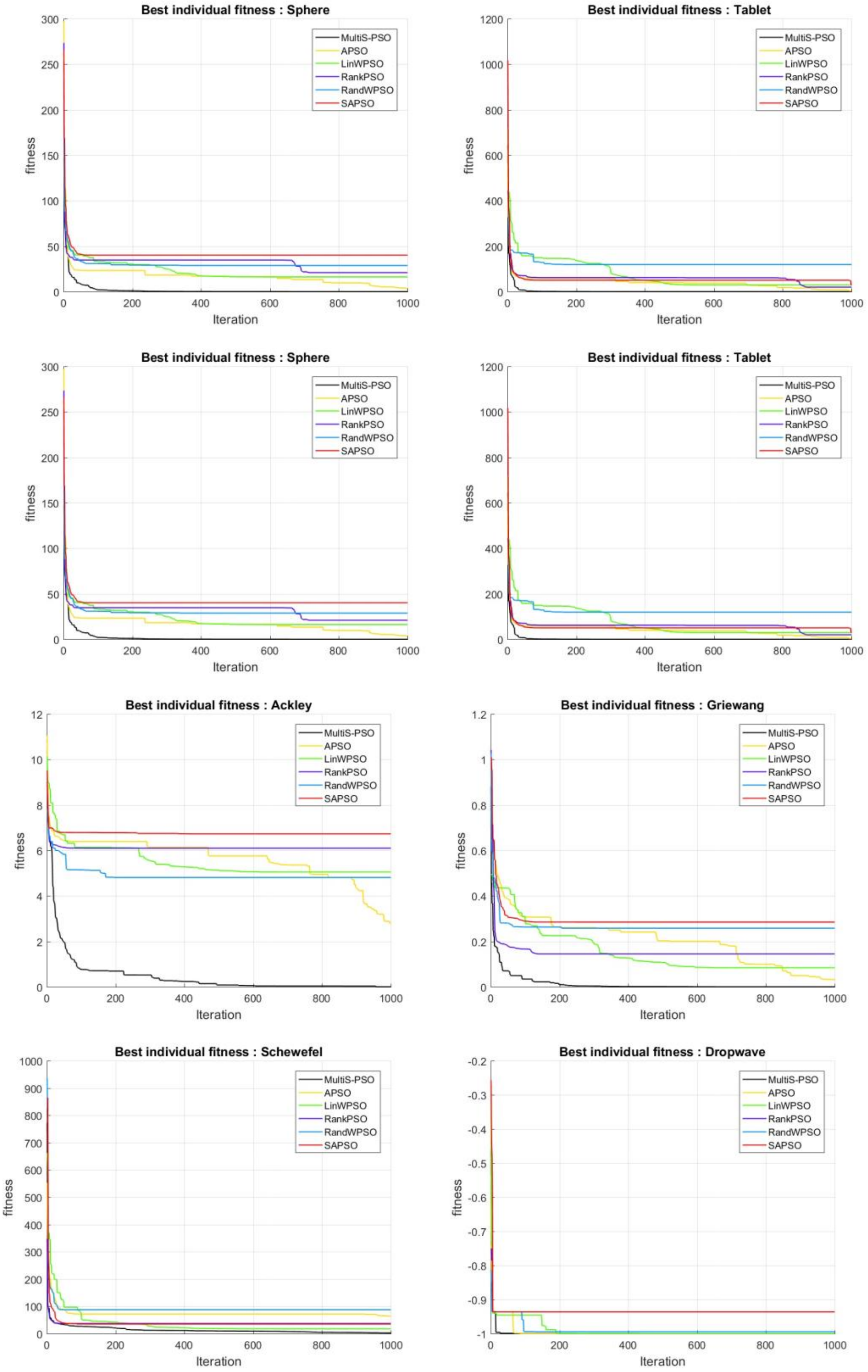


Figure 3 Comparison on convergence speed (Continued)



As shown in figure 3, the black line that gives the executions of MultiS-PSO results is under all other lines. Subsequently, MultiS-PSO provides a quicker convergence when compared to all diverse used PSO variants from literature.

Given the exposed effective results of our approach, we can most certainly recognize that the multi-swarm cooperation based on hidden markov model supplies noticeably more significant performances for the PSO algorithm regarding the solution accuracy and the convergence speed.

5 Conclusion

As a conclusion, we have displayed a new approach named MultiS-PSO that uses a multi swarm design based on a hidden markov model with a master/slave cooperation rule. Each one of PSO particles uses its historical information and its current swarm to choose the next swarm which it will belong. Our multi swarm approach is powered by an attached hidden Markov chain to each element of the swarm that provides swarm control of particle during the search process. According to each swarm, acceleration coefficients are updated. Then, the cooperation between swarms boost more the search. Experimental results have established very competitive performances in comparison to several chosen PSO variants. We can deduce from obtained results that associating a multi swarm based machine learning with a cooperation strategy enhances significantly PSO performances.

References

[1] Oussama Aoun, Malek Sarhani, and Abdellatif El Afia. Hidden markov model classifier for the adaptive particle swarm optimization. In *The XI Metaheuristics International Conference (MIC)*, Agadir, Morocco, 2015.

[2] Mehmet Emin Aydin. Coordinating metaheuristic agents with swarm intelligence. *Journal of Intelligent Manufacturing*, 23(4):991–999, 2012.

[3] Benjamin Bengfort, Philip Y Kim, Kevin Harrison, and James A Reggia. Evolutionary design of self-organizing particle systems for collective problem solving. In *Swarm Intelligence (SIS), 2014 IEEE Symposium on*, pages 1–8. IEEE, 2014.

[4] Gerardo Beni. From swarm intelligence to swarm robotics. In *Swarm robotics*, pages 1–9. Springer, 2004.

[5] Tim Blackwell, Jürgen Branke, et al. Multi-swarm optimization in dynamic environments. In *EvoWorkshops*, volume 3005, pages 489–500. Springer, 2004.

[6] Manuele Brambilla, Eliseo Ferrante, Mauro Birattari, and Marco Dorigo. Swarm robotics: a review from the swarm engineering perspective. *Swarm Intelligence*, 7(1):1–41, 2013.

[7] Marco A Montes De Oca, Thomas Stützle, Ken Van den Enden, and Marco Dorigo. Incremental social learning in particle swarms. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 41(2):368–384, 2011.

[8] Dagmar Monett Diaz. *Agent-Based Configuration of (Metaheuristic) Algorithms*. PhD thesis, Humboldt University of Berlin, 2005.

[9] Marco Dorigo and Luca Maria Gambardella. Ant colony system: a cooperative learning approach to the traveling salesman problem. *Evolutionary Computation, IEEE Transactions on*, 1(1):53–66, 1997.

[10] Pierre Dupont, François Denis, and Yann Esposito. Links between probabilistic automata and hidden markov models: probability distributions, learning models and induction algorithms. *Pattern recognition*, 38(9):1349–1371, 2005.

[11] Weichang Jiang, Yating Zhang, and Ruihua Wang. Comparative study on several pso algorithms. In *Control and Decision Conference (2014 CCDC), The 26th Chinese*, pages 1117–1119, May 2014.

[12] Hu Jianxiu and Zeng Jianchao. A two-order particle swarm optimization model [j]. *Journal of Computer Research and Development*, 11:004, 2007.

[13] J. Kennedy and R.C . Eberhart. Particle swarm optimization. *Proceedings of IEEE international conference neural networks*, IEEE:1942–8, 1995.

[14] Changhe Li, Shengxiang Yang, and Trung Thanh Nguyen. A self-learning particle swarm optimizer for global optimization problems. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 42(3):627–646, 2012.

[15] Junliang Li and Xinping Xiao. Multi-swarm and multi-best particle swarm optimization algorithm. In *Intelligent Control and Automation, 2008. WCICA 2008. 7th World Congress on*, pages 6281–6286. IEEE, 2008.

[16] Jing J Liang, A Kai Qin, Ponnuthurai Nagaratnam Suganthan, and S Baskar. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. *Evolutionary Computation, IEEE Transactions on*, 10(3):281–295, 2006.

[17] Li-li LIU and Xing-bao GAO. An adaptive simulation of bacterial foraging algorithm. *Basic Sciences Journal of Textile Universities*, 4:022, 2012.

[18] Yu Liu, Zheng Qin, Zhewen Shi, and Jiang Lu. Center particle swarm optimization. *Neurocomputing*, 70(4-6):672–679, 2007.

[19] Seyedali Mirjalili, Andrew Lewis, and Ali Safa Sadiq. Autonomous particles groups for particle swarm optimization. *Arabian Journal for Science and Engineering*, 39(6):4683–4697, 2014.

[20] Ben Niu, Yunlong Zhu, Xiaoxian He, and Henry Wu. Mcpso: A multi-swarm cooperative particle swarm optimizer. *Applied Mathematics and computation*, 185(2):1050–1062, 2007.

[21] Abdellatif El Afia Oussama Aoun, Malek Sarhani. Particle swarm optimization with population size and acceleration coefficients adaptation using hidden markov model state classification. *International Journal of Metaheuristics*, in press.

[22] L. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, Feb 1989.

[23] Malek Sarhani and Abdellatif El Afia. Facing the feature selection problem with a binary pso-gsa approach. In *The XI Metaheuristics International Conference (MIC)*, Agadir, Morocco, 2015.

[24] Yang Shi, Hongcheng Liu, Liang Gao, and Guohui Zhang. Cellular particle swarm optimization. *Inf. Sci.*, 181(20):4460–4493, October 2011.

[25] Frans van den Bergh and Andries Engelbrecht. A cooperative approach to particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, 8(3):225–239, 2004.

[26] Shengsheng Wang, Mo Chen, Dezhi Huang, Xu Guo, and Chuangfeng Wang. Dream effected particle swarm optimization algorithmâ‹†.

[27] Z Wu. Optimization of distribution route selection based on particle swarm algorithm. *International Journal of Simulation Modelling (IJSIMM)*, 13(2), 2014.

[28] S. Yang and C. Li. A clustering particle swarm optimizer for locating and tracking multiple optima in dynamic environments. *IEEE Transactions on Evolutionary Computation*, 2010.

[29] Zhi-Hui Zhan, Jun Zhang, Yun Li, and H.S.-H. Chung. Adaptive Particle Swarm Optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 39(6):1362–1381, December 2009.

[30] Jiuzhong Zhang and Xueming Ding. A multi-swarm self-adaptive and cooperative particle swarm optimization. *Engineering applications of artificial intelligence*, 24(6):958–967, 2011.

Exact and multi-objective evolutionary-based approaches for process plan generation in a reconfigurable manufacturing environment

Faycal A. Touzout[1], Lyes Benyoucef[1], Hichem Haddou Benderbal[1] and Mohamed Dahane[2]

[1] LIS UMR 7020, Aix-Marseille University, Marseille, France
{faycal.touzout,lyes.benyoucef,hichem.haddoubenderbal}@lis-lab.fr
[2] LGIPM Research Laboratory, Lorraine University, Metz, France
mohamed.dahane@univ-lorraine.fr

Abstract. Low costs, high reactivity and high quality products are necessary criteria for industries to achieve competitiveness in nowadays market. In this paper, three approaches are proposed and compared for the process plan generation problem in a reconfigurable manufacturing environment: an iterative multi-objective integer linear program (I-MOILP) and adapted versions of the well-known evolutionary algorithms, respectively, archived multi-objective simulated annealing (AMOSA) and the non-dominated sorting genetic algorithm (NSGA-II). Moreover, in addition to the minimization of the classical total production cost and the total completion time, the minimization of the maximum machines exploitation time is considered as a novel optimization criterion, in order to have high quality products. To illustrate the applicability of the three approaches, an example is presented and the obtained numerical results are analysed.

Keywords: *reconfigurable manufacturing system, multi-objective optimization, multi-objective integer linear programming, AMOSA, NSGA-II.*

1 Introduction

With burgeoning global markets and the demanding nature of the customer, it is very important for companies/organizations to respond quickly and cost effectively to be present and to take the lead among the competitors. Today, customer satisfaction is a challenge for most manufacturing companies. Mass customization, a product deployment concept that combines low price with extensive variation and adaptation has emerged due to its potential impact upon the customer regarding the perceived value of the product.

With the continuous demand for products incorporating new and complex functionalities there has been a lot of pressure on the manufacturing companies. This requires a changeable structure of the organization to cater to a wide product variety and can be attained through adoption of the concept of Reconfigurable Manufacturing System (RMS), which comprises of reconfigurable machines, controllers and the software support systems.

RMS was proposed during the mid-nineties by *Koren*. Due to its six key characteristics: *modularity, integrability, customization, convertability, scalability, and diagnosability* [1], it is thought to be one of the most suitable paradigms to answer these challenges. *However, planning, managing and optimizing, in this context, are an exponentially more complex tasks.*

In this paper, three approaches are proposed and compared for the process plan generation problem in a reconfigurable manufacturing environment: an iterative multi-objective integer linear program (I-MOILP) and adapted versions of the well-known evolutionary algorithms, respectively, archived multi-objective simulated annealing (AMOSA) and the non-dominated sorting genetic algorithm (NSGA-II). Moreover, in addition to the minimization of the classical total production cost and the total completion time, the minimization of the maximum machines exploitation time is considered as a novel optimization criterion, in order to have high quality products.

The rest of the paper is organized as follows: Section 2 briefly summarizes the related works to RMS. Section 3 presents the problem under consideration and its mathematical formulation. Section 4 describes more in details the proposed three approaches. Section 5 analyses the obtained numerical results. Section 6 concludes the paper with some future research directions.

2 Literature review

The literature related to RMS problems is very rich and covers many areas, such as: design, layout optimization, reconfigurable control, process planning and production scheduling [2]. However, in this section, we will summarize the most related research works to process plan generation in RMS.

Azab et ElMaraghy [3] considered reconfigurable process plans, where an existing process plan is reconfigured when a new feature is added to an existing part, in order to avoid the generation of a wholly new process plan. Reconfiguration of process plan consists to include minor modifications to meet the requirements of the new part. Furthermore, *Shabaka et ElMaraghy* [4] developed a new genetic algorithm based model to perform process plan in RMS environment. The model simultaneously considers all process plan parameters such as machine assignment and machine configurations.

Bensmaine et al. [5] addressed the problem of process plan generation in RMS from a multi-objective perspective using an adapted version of AMOSA. They elaborated an experimental comparison based on the obtained pareto fronts. In the same direction, *Musharavati et al.* [6] used an enhanced simulated-annealing-based approach. The enhancement of the proposed approach was conducted by combining variants of the simulated-annealing technique with other algorithm concepts such as knowledge-exploitation and parallelism. While, *Chaube et al.* [7] presented an adapted version of the NSGA-II. Two objective are considered, respectively, the total completion time and manufacturing cost.

More recently, *Haddou Benderbal et al.* [8] adapted AMOSA to solve the integrated design and process plan generation problem for RMS. In addition to the classical optimization criteria, respectively, cost and time, the authors added modularity as a criterion. *Xia et al.* [9] extended the concept of reconfigurable process planning to a concept of reconfigurable machining process planning which targets the process plan generation for a part family.

3 Problem description and mathematical formulation

3.1 Problem description

The manufacturing of a product is the processing of a set of operations linked with each other by a precedence graph (e.g. Fig. 1). The problem of generating a process plan is to define a sequencing of these operations for a given RMS's design (i.e. a set of machines, configurations and tools).

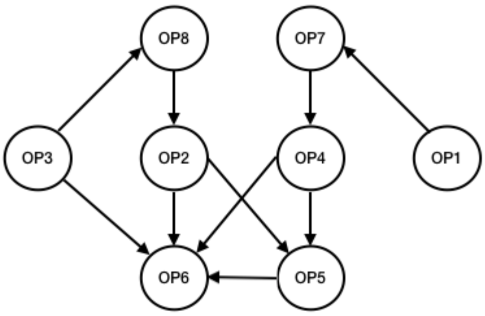


Fig. 1. An example of a precedence graph

A machine in a RMS is represented by a set of available configurations and a set of compatible tools. A configuration in this case offers different tool advance directions (TADs) (i.e. $x_\pm$, $y_\pm$ and $z_\pm$). An operation is represented by the TADs that it requires to be processed. Thus, a set of triplets K_i that are able to perform the operation i is defined. A triplet k in this case is defined by the indices of its machine $indexM^k$, configuration $indexC^k$ and tool $indexT^k$.

Table 1 shows an example of the required TADS and tools for the operations of an instance of the problem. The TADS, configurations and tools that each machine offers are also presented. Table 2 illustrates an example of a generated process plan.

In this case, three optimization criteria are considered: (i) The total production cost, (ii) The total completion time, (iii) The exploitation time per machine

Table 1. An illustrative example of TADS and Tools

OPs Ms | Cs		x_+	y_+	z_+	x_-	y_-	z_-	Tools
OP1		x			x			4
OP2		x	x				x	3
OP3			x					4
OP4		x			x			4
OP5		x		x		x		2
OP6		x	x					4
OP7			x					2
OP8		x						2
M1	C1	x			x			3,4
	C2	x		x		x		
M2	C1	x		x		x		2,3,4
	C2			x		x		
M3	C1	x			x			3,4
	C2			x	x	x		
M4	C1	x						2,3
	C2			x	x	x		
	C3				x		x	
	C4	x	x		x		x	
M5	C1	x		x		x		2,3,4
	C2	x	x			x	x	

Table 2. An illustrative example of a process plan

Operation	OP1	OP7	OP3	OP8	OP2	OP4	OP5	OP6
Machine	M3	M4	M5	M5	M5	M1	M2	M4
Config	C1	C4	C2	C2	C2	C1	C1	C2
Tool	T4	T2	T4	T2	T3	T4	T2	T4

3.2 Mathematical Formulation

Throughout the next section, the following notations are used:

Parameters :

O : Set of operations
n : Number of operations
i, i' : Index of operation
P_i : Set of predecessors operations
k, k' : Index of triplet

j, j' : Index of position in the sequence
M : Set of machines
m, m' : Index of machine
K_i : Set of available triplets for operation i
K_m : Set of available triplets with machine m
K : Set of triplets, where: $K = K_i \cup K_m$
h, h' : Index of configuration
tl, tl' : Index of tool

Production costs :
$CCM_{m,m'}$: Cost of changing machine per time unit
$CCC_{h,h'}$: Cost of changing configuration per time unit
$CCT_{tl,tl'}$: Cost of changing tool per time unit
$CP_{i,k}$: Cost of processing per time unit

Time :
$TCM_{m,m'}$: Time of changing machine
$TCC_{h,h'}$: Time of changing configuration
$TCT_{tl,tl'}$: Time of changing tool
$TP_{i,k}$: Time of processing

To formulate our problem, these decisions variables are needed:

$x_{i,j}^k = 1$ if the i^{th} operation is processed at the j^{th} position using the k^{th} triplet, 0 otherwise.

$y_{j,k}^m = 1$ if the m^{th} machine is using the k^{th} triplet at the j^{th} position, 0 otherwise.

$cm_{j,m,m'} = 1$ if between position $j-1$ and j, there has been a change between machines m and m', 0 otherwise.

$cc_{j,k,k'}^m = 1$ if between position $j-1$ and j, there has been a change between triplet k and k' of machine m, 0 otherwise.

$f^e \in \mathbb{N}$ represents the maximal exploitation time of the machines.

f^c and f^t are, respectively, the total production cost and the completion time, where:

$$f^c = \sum_{j=1}^{n} \sum_{i \in O} \sum_{k \in K_i} x_{i,j}^k \times CP_{i,k} \times TP_{i,k} + \sum_{j=1}^{n} \sum_{m \in M} \sum_{m' \in M} cm_{j,m,m'} \times CCM_{m,m'} \times TCM_{m,m'} + \sum_{j=1}^{n} \sum_{m \in M} \sum_{k \in K^m} \sum_{k' \in K^{m'}} cc_{j,k,k'}^m \times (TCT_{indexT^k,indexT^{k'}} \times CCT_{indexT^k,indexT^{k'}} + TCC_{indexC^k,indexC^{k'}} \times CCC_{indexT^k,indexT^{k'}})$$

$$f^t = \sum_{j=1}^{n} \sum_{i \in O} \sum_{k \in K_i} x_{i,j}^k \times TP_{i,k} + \sum_{j=1}^{n} \sum_{m \in M} \sum_{m' \in M} cm_{j,m,m'} \times TCM_{m,m'} + \sum_{j=1}^{n} \sum_{m \in M} \sum_{k \in K^m} \sum_{k' \in K^{m'}} cc_{j,k,k'}^m \times (TCC_{indexC^k,indexC^{k'}} + TCT_{indexT^k,indexT^{k'}})$$

Our problem can be formulated as a Multi-Objective Integer Linear Program (MOILP).

Constraint (1) states that one operation is processed at each position of the process plan. Constraint (2) states that each operation is processed once. Constraint (3) states that an operation is processed if and only if all its predecessors operations are already processed. Constraint (4) states that each machine is using one configuration and one tool at once. Constraint (5) states which configuration and tool are used at position j for machine m. Constraints (6) and (7) state, respectively, if there's a change of machine and a change of configuration and/or tool between positions $j-1$ and j. Constraint (8) states that there's only one change of configuration between positions $j-1$ and j. Finally, constraint (9) states the maximal exploitation time.

MOILP

$min\ f^c$
$min\ f^t$
$min\ f^e$

$s.t.$

$$\begin{array}{llll}
\sum_{i\in O}\sum_{k\in K_i} x_{i,j}^k & = 1 & \forall j = 1...n & (1) \\
\sum_{j=1}^{n}\sum_{k\in K_i} x_{i,j}^k & = 1 & \forall i \in O & (2) \\
\sum_{k\in K_i} x_{i,j}^k \times |P_i| & \leq \sum_{i'\in P_i}\sum_{j'=1}^{j-1}\sum_{k'\in K_{i'}} x_{i',j'}^{k'} & \forall i \in O, \forall j = 1...n & (3) \\
\sum_{k\in K_m} y_{j,t}^m & = 1 & \forall j = 1...n, \forall m \in M & (4) \\
y_{j,k}^m & \geq x_{i,j}^k & \forall j = 1...n, \forall m \in M, \forall k \in K_m & (5) \\
\sum_{i\in O} x_{i,j}^k + x_{i,j-1}^{k'} & \leq ct_{j,indexM^k,indexM^{k'}} + 1 & \forall j = 2...n, \forall k, k' \in K & (6) \\
y_{j,k}^m + y_{j-1,k'}^m & \leq cc_{j,k,k'}^m + 1 & \forall j = 2...n, \forall m \in M, \forall k, k' \in K_m & (7) \\
\sum_{k,k'\in K_m} cc_{j,k,k'}^m & = 1 & \forall j = 1...n, \forall m \in M & (8) \\
\sum_{k\in K_m}\sum_{i,j\in O} x_{i,j,k} \times TP_{i,k} & \leq f^e & \forall m \in M & (9)
\end{array}$$

4 Proposed Approaches

In this section, we will describe in details the three developed approaches.

4.1 Iterative Multi-Objective Integer Linear Program (I-MOILP)

I-MOILP is a cutting-plane based approach that enumerates the whole optimal pareto front. The idea behind is to solve at each iteration $iter$ a smaller integer linear program (ILP) by adding cuts (i.e. constraints) to eliminate the efficient solution generated by the ILP at $iter-1$ as well as all the solutions dominated by it from the search space.

A description of our approach is proposed in Algo. 1, where M_k is an upper-bound of the k^{th} objective, z_k^{iter} a boolean variable used to select the objective to optimize at iteration $iter$ and da the minimum dispersion amount between two solutions for an objective. It is important to note that our Iterative-MOILP can be used to tackle other multi-objective optimization problems. However, the problem must be modelled as an integer linear program.

Although the enumeration of the whole optimal pareto front by I-MOILP for large-sized instances can be a very time consuming task, it still has undeniable advantages:

Algorithm 1: Iterative-MOILP

```
input data
$iter = 0$
set an empty archive
aggregate the objectives of MOILP
solve MOILP
add solution to the archive
while MOILP is still feasible do
    $iter++$
    $expr = 0$
    create variables $z_k^{iter}$
    for $k = 1...nbObjectives$ do
        $expr = expr + z_k^{iter}$
        add constraint: $f_k \leq (f_k^{iter-1} - da)z_k^{iter} + M_k(1 - z_k^{iter})$
    end for
    add constraint: $expr \geq 1$
    solve MOILP
    add solution to the archive
end while
return archive
```

(i) It can provide the exact number of solutions asked by the decision maker (DM)

(ii) It can control the dispersion of the provided solutions by manipulating the dispersion amount *da*

(iii) It provides the most appealing solutions to the DM from the beginning when the weights of the objectives are properly defined

4.2 Adapted Archived Multi-Objective Simulated-Annealing (AMOSA)

AMOSA is a simulated annealing based multi-objective optimization algorithm that provides a set of solutions non-dominated with each other for a considered problem. Starting with a randomized or a given initial solution, a local search is performed to generate a new one. An elaborate procedure is used to determine the acceptance of the new solution in the archive.

A brief description of our approach is proposed in Algo. 2. [10] proposes a more detailed description as well as a complexity study and a comparison with the well-known evolutionary algorithms NSGA-II and PAES for well-known problems in the literature.

Algorithm 2: Adapted AMOSA

```
input data
initialize $Tmax, Tmin, iter, \alpha, temp = Tmax$, $perturbationRatio$ archive
current=random(archive)
while $temp > Tmin$ do
    for $i = 0 : iter$ do
        new=perturb(current)
        depending on the dominance status of new with current and the solutions in the archive: new
        replace current and is added to the archive
        delete dominated process plans from the archive
    end for
    $temp = \alpha \times temp$
end while
return archive
```

4.3 Adapted Non Dominated Sorting Genetic Algorithm II (NSGA-II)

The Non Dominated Sorting Genetic Algorithm (NSGA-II) is a population-based evolutionary algorithm proposed by [11]. Starting with a randomized initial population called the *parent population* of a given size, for each iteration of NSGA-II, a new population called *child population* is generated by applying genetic operators (i.e. mutation, crossovers...) with specified probabilities. The parent population of iteration $iter + 1$ is the result of an elitist procedure applied to $parentPopulation_{iter} \cup childPopulation_{iter}$. This elitist procedure is ensured by a fast non dominated sorting algorithm, as well as a crowding distance sorting.

A description of our approach is proposed in Algo. 3. Moreover, more detailed descriptions of the fast non dominated sorting and the crowding distance sorting algorithms are presented in [11].

Algorithm 3: Adapted NSGA-II

```
input data
initialize $populationSize$, $iteration$, $p_{mutation}$, $mutationRatio$, $p_{crossover}$
randomize $parentPopulation$
for $iter = 1 : iteration$ do
    generate $childPopulation$ from $parentPopulation$
    $population = parentPopulation \cup childPopulation$
    $F = fastNonDominatedSorting(population)$
    for $l = 1 : size(F)$ do
        if size($newPopulation$)+size($F_l$) $<$ $populationSize$ then
            $newPopulation += F_l$
        else
            $crowdingDistanceSorting(F_l)$
            for $k = 1 : size(F_l)$ do
                if size($newPopulation$) $<$ $populationSize$ then
                    $newPopulation += F_l^k$
                else
                    break;
                end if
            end for
        end if
    end for
    $parentPopulation = newPopulation$
end for
return $parentPopulation$
```

5 Experimental results and analyses

Due to the lack of benchmarks in the literature related to process plan generation in a reconfigurable manufacturing environment, our experiments results are performed with randomly generated instances. An instance is identified by the number of operations and the number of machines and denoted by *nbOperations_nbMachines*.

In order to study the influence of the probabilities of genetic operators on the convergence of adapted NSGA-II, we tested various versions. A version in our case is represented by the probability of its mutation operator (e.g. NSGA-90 is an adapted version of NSGA-II where 90 % of the child population is generated using mutation operations while the rest is the result of crossover operations). For visualization purposes, we limited our results to 5 versions of the adapted NSGA-II, which are presented in the comparisons.

In this case, we can distinguish two experimental schemes:

5.1 Experimental scheme 1

Adapted AMOSA and adapted NSGA-100, NSGA-75, NSGA-50, NSGA-25 and NSGA-0 are executed 10 times and the average percentage of the number of solutions in the optimal pareto front is compared in regards to the obtained pareto front from the use of I-MOILP. Table 3 presents the obtained results for instances of the problem under consideration. The computational time regarding 10_5 for the I-MOILP is of 438 seconds.

Table 3. Averages percentages of the number of solutions in the optimal pareto fronts: I-MOILP vs adapted AMOSA and NSGA-II

Instance	I-MOILP	AMOSA	NSGA-100	NSGA-75	NSGA-50	NSGA-25	NSGA-0
3_3	100 %	100 %	100 %	100 %	100 %	100 %	100 %
4_3	100 %	100 %	100 %	100 %	100 %	100 %	100 %
5_3	100 %	100 %	100 %	100 %	100 %	100 %	100 %
6_4	100 %	54.8 %	100 %	100 %	100 %	100 %	52.8 %
7_5	100 %	35 %	100 %	100 %	100 %	100 %	25 %
8_5	100 %	55 %	100 %	100 %	100 %	98 %	61.2 %
9_5	100 %	100 %	100 %	100 %	100 %	100 %	90 %
10_5	100 %	0 %	95 %	85 %	92.5 %	70 %	2.5 %

As we can see from Table 3, adapted NSGA-II outperforms adapted AMOSA. We observe that the pareto fronts obtained by using the adapted NSGA-II are, in most cases, *optimal.* Moreover, we realise that for these small instances, when the crossover operator is used exclusively, the quality of the pareto fronts deteriorates.

5.2 Experimental scheme 2

For medium and large-sized instances, the enumeration of the whole optimal pareto fronts by I-MOILP can be very time consuming or even impossible. In this case, comparisons between the generated pareto fronts of, respectively, adapted AMOSA and adapted NSGA-100, NSGA-75, NSGA-50, NSGA-25 and NSGA-0 are presented in Figures 2, 3 and 4.

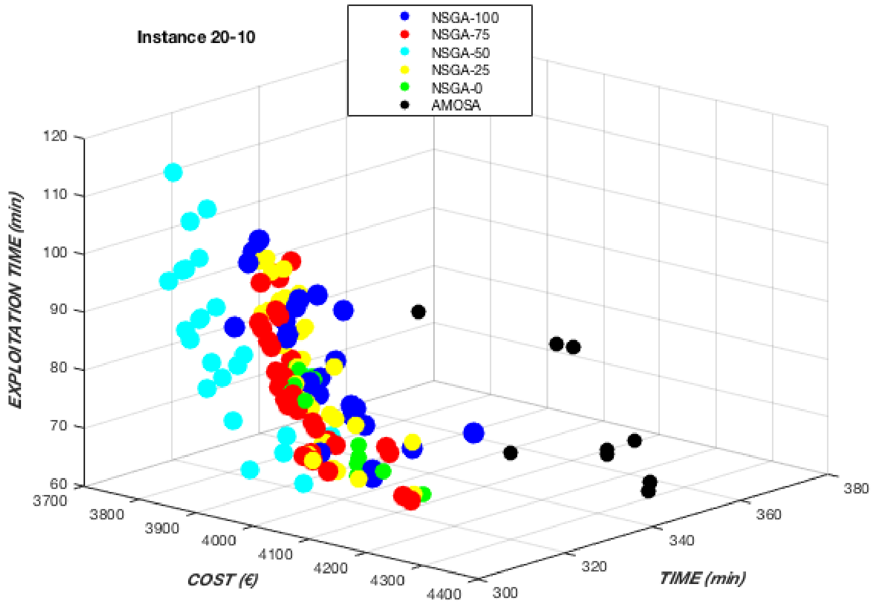


Fig. 2. Pareto fronts of instance 20_10: AMOSA vs NSGA-II

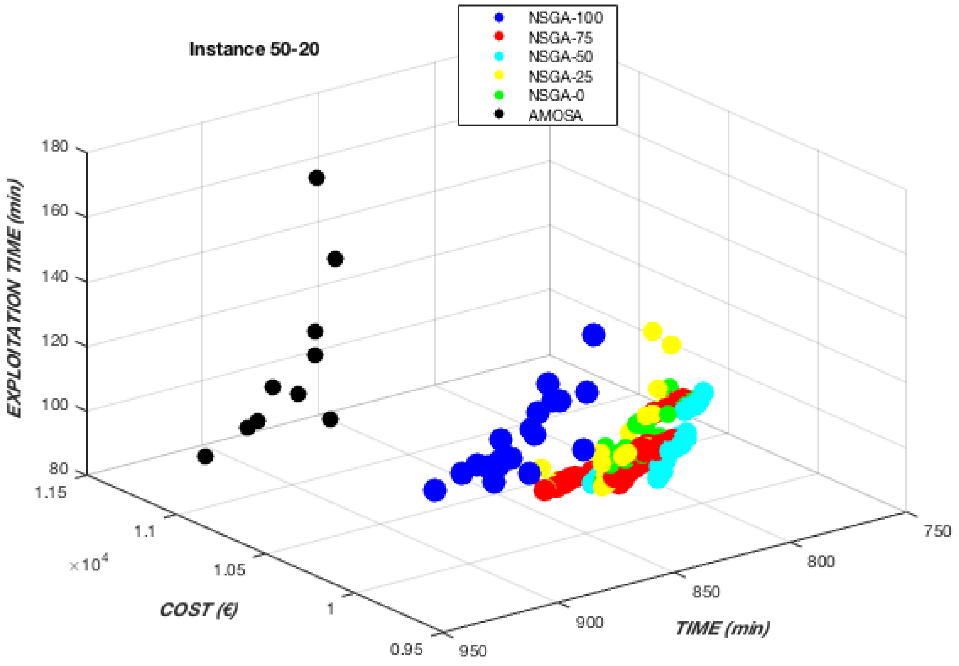


Fig. 3. Pareto fronts of instance 50_20: AMOSA vs NSGA-II

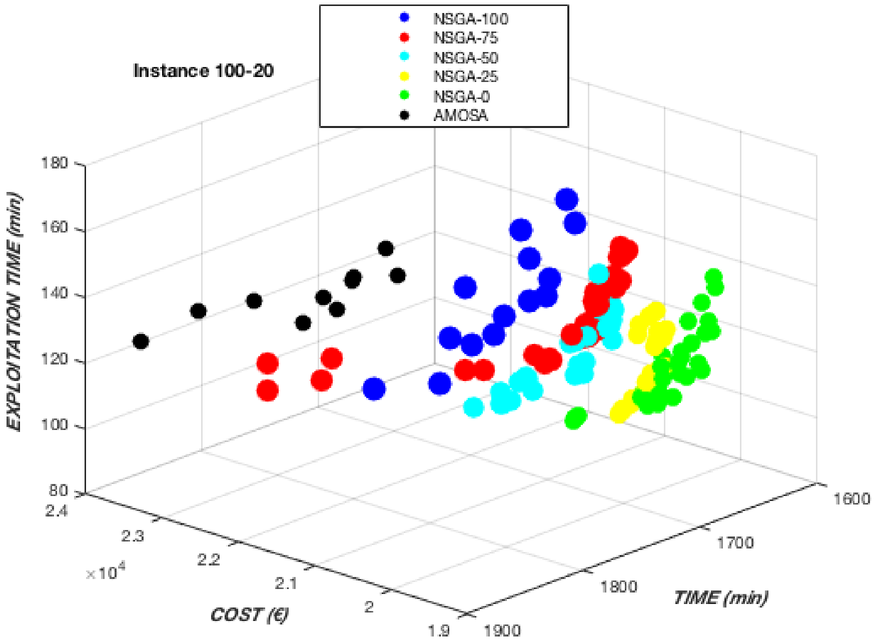


Fig. 4. Pareto fronts of instance 100_20: AMOSA vs NSGA-II

Confirmed by our experiments, we can observe that the adapted NSGA-II still performs better than the adapted AMOSA for larger instances. Moreover, we can see that the larger the instance is, the more efficient the crossover operator gets, and vice-versa. Finally, we can claim that the inefficiency of the mutation operator for large-sized instances is the result of the large number of neighbours of a given solution.

6 Conclusions

In this paper, we presented and compared an exact and two multi-objective evolutionary-based approaches for the process plan generation problem in RMS. Three criteria were considered, respectively, the total production cost, the completion time and the maximum exploitation time for machines. We presented a rich panel of experimental results and analyses to demonstrate the quality of the developed three approaches.

For future works, we expect to use other criteria such as greenhouse gas (GHG) emission for sustainability purposes. Moreover, we consider extending the problem to two variants such as, the multi-unit process plan generation problem and the integrated process plan generation and scheduling problem (IPPS) in reconfigurable environment.

References

1. Koren, Y. (2006). General RMS characteristics. Comparison with dedicated and flexible systems. In Reconfigurable manufacturing systems and transformable factories (pp. 27-45). Springer, Berlin, Heidelberg.
2. Bensmaine, A., Dahane, M., & Benyoucef, L. (2014). A new heuristic for integrated process planning and scheduling in reconfigurable manufacturing systems. International Journal of Production Research, 52(12), 3583-3594.
3. Azab, A., & ElMaraghy, H. A. (2007). Mathematical modeling for reconfigurable process planning. CIRP Annals-Manufacturing Technology, 56(1), 467-472.
4. Shabaka, A. I., & ElMaraghy, H. A. (2008). A model for generating optimal process plans in RMS. International Journal of Computer Integrated Manufacturing, 21(2), 180-194.
5. Bensmaine, A., Dahane, M., & Benyoucef, L. (2012). Process plan generation in reconfigurable manufacturing systems using AMOSA and TOPSIS. IFAC Proceedings Volumes, 45(6), 560-565.
6. Musharavati, F., & Hamouda, A. S. M. (2012). Enhanced simulated-annealing-based algorithms and their applications to process planning in reconfigurable manufacturing systems. Advances in Engineering Software, 45(1), 80-90.
7. Chaube, A., Benyoucef, L., & Tiwari, M. K. (2012). An adapted NSGA-2 algorithm based dynamic process plan generation for a reconfigurable manufacturing system. Journal of Intelligent Manufacturing, 23(4), 1141-1155.
8. Benderbal, H. H., Dahane, M., & Benyoucef, L. (2018). Modularity assessment in reconfigurable manufacturing system (RMS) design: an Archived Multi-Objective Simulated Annealing-based approach. The International Journal of Advanced Manufacturing Technology, 94(1-4), 729-749.
9. Xia, Q., Etienne, A., Dantan, J. Y., & Siadat, A. (2018). Reconfigurable machining process planning for part variety in new manufacturing paradigms: Definitions, models and framework. Computers & Industrial Engineering, 115, 206-219.
10. Bandyopadhyay, S., Saha, S., Maulik, U., & Deb, K. (2008). A simulated annealing-based multiobjective optimization algorithm: AMOSA. IEEE transactions on evolutionary computation, 12(3), 269-283.
11. Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. A. M. T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE transactions on evolutionary computation, 6(2), 182-197.

A new heuristic method for a dynamic pricing and production problem

Paulin Couzon, Yassine Ouazene and Farouk Yalaoui

Industrial Systems Optimization Laboratory, University of Technology of Troyes
12 rue Maris Curie, CS 42060, 10004 Troyes, France
{paulin.couzon;yassine.ouazene;farouk.yalaoui}@utt.fr

1 Introduction

The dynamic pricing, in the case of a manufacturing system allows companies to better manage their stocks, their production and permits to increase their profits. In a low inventory situation, the manager can choose to increase the product's price to reduce the demand and prevent costly out-of-stock penalties. In the same way, the manager can reduce the price to raise the demand and use more efficiently its production resources in a high production capacity situation.

From a historical point of view, before the early 80s, the dynamic pricing had mostly been studied by economists. A few papers from the lot-sizing community dealing with the coordinated pricing and production problem ([1]) was published. In 1978, the Airline Deregulation Act has been voted in the USA. This law opened the airline industry market to new companies and allowed them to set their own prices. This led the market to be hyper competitive, and the companies had to change their revenue and seat management. That is the beginning of the "yield management", that helped companies to make profit but also to survive ([2]).

Following the success of the yield managment, there have been a revival of interest towards dynamic pricing by the research community in the mid 90s, but it was mostly focused on systems with only inventory decisions. For the production side, Haugen et al. [4] studied a lot-sizing problem combined with pricing aspects. However, the authors considered only the case of linear demand function. Ahmadi and Shavandi [5] addressed dynamic pricing in a production system with a single product, demanded by several customer classes. They found the structure of the optimal policy for two pricing conditions. Some other papers such as Adida and Perakis [6] used also game theory approach to model the competition between companies.

The main issue of these papers and more generally of most of the literature publication is the lack of real world application studies. In fact, thanks to the development of the internet, the change of price for the e-retailers is less expensive than the change of price in "brick and mortar" store. However, it's still difficult to do extensive tests to modelize the impact of price on customers. Fisher et al. [7] illustrate the difficulty of "demand learning" in an inventory problem by conducting a field experiment with a chinese retailer.

The objective of our research is to propose a general model, which can be adapted to different real case applications by changing the input data. A new heuristic method to solve the problem of coordinating production operations and pricing strategies is proposed. This method is tested based on some instances from the literature. The obtained results are promising.

2 Optimization approach

2.1 Initial model

The model considered in this work is based on a problem addressed by Bajwa et al. [9]. Their mathematical formulation is a modification of a classical lot-sizing formulation with setup costs. The problem represents a firm that can produce and stock different products. These products prices and demands are independants. There is no competitors in the market, and the firm wants to maximize its profit. At each period, the firm has to decide how much to produce, how much to stock, how much to sell and at which price.

The following assumptions are considered: the set of products is fixed, the time horizon is discretized into periods. The demand of a customer for a product is formalized by a function which depends on the product's price, seasonality and maximum capacity parameters. In our case, two well-known demand functions, linear and isoelastic, are considered.

The authors used an adaptation of the Outer Approximation Algorithm (OA) to solve their model. OA is an exact algorithm based on a decomposition into a primal model and a master model. It was initially introduced by Duran et al. [8] to solve a nonlinear problem. The model proposed by Bajwa et al. [9] is well adapted to the Outer-approximation methology because the binary variables of the model don't affect the feasibility of the solution. Therefore, there is no need to check the feasibility of the solutions during the process. This method is able to solve small instances, but as most of the exact algorithms, it becames less efficient with the increasing of the instances' size.

2.2 Heuristic implemented

Since the addressed problem is known to be NP-hard, a heuristic method is proposed to solve large instances in reasonable computational time. The decision of implementing a heuristic method to solve the problem has been made because of the increasing of computation time as the instances grow in size. The purpose of the heuristic is to get one or several near-optimal solutions. This method replaces the resolution of the master problem. It gives a setup configuration to the primal problem, but it doesn't provide an upper bound to the objective function. Then the primal problem is solved optimally by the efficient algorithm proposed by Bajwa et al. [9].

The heuristic method is based on two constructive heuristics and different local search moves. For the first heuristic, the products are initially sorted following a given decision rule. Then, the setup values of one product are set before the decision of the next one. Finally, the obtained solution is evaluated. The second heuristic operates differently. At each period, the algorithm decides which products to produce, and for how many periods. In order to determine the best choice, the algorithm evaluates its partial solution value. For the evaluation, the remaining non decided periods are filled with setups and the resolution of the primal model provides the value. Thanks to this partial evalution, the algorithm is able to choose locally the best solution. To select which product setup should be assigned, the two heuristics consider that a production can begin only when the stock level is equal to zero at the end of the previous period (Wagner-Whitin heuristic).

The local search moves are applied to the best solution obtained by the constructive heuristics to improve it. These moves aim to decrease the number of setup, exchange the setup for two products and swap the setup of a product. They are implemented with a "first improvement" policy.

2.3 Numerical experiments

In this section, some promising preliminary results obtained are presented. The instances proposed by Bajwa et al. [9] are used as benchmark to test the heuristic. The obtained results are reported in table 1 for linear demand function and table 2 for isoelactic demand function. The column H1 represents the best result for the first heuristic among the different sorting rules. The column H2 represents the result for the second heuristic. The RL columns provide the results of the local search used after the constructive heuristic. Only the gap between the solution given by the method based on Outer approximation for the linear function and on Lingo solver for the isoelastic function, and the solution given by the heuristic method is presented.

Table 1. Linear demand, decreasing seasonality over time

Production capacity	Exact method	H1	H1+RL	H2	H2+RL
30	235.7	7.7%	0.0%	0.0%	0.0%
40	258.5	7.7%	0.6%	0.0%	0.0%
50	264.5	2.5%	0.0%	0.8%	0.0%
60	267.8	2.6%	1.4%	4.5%	4.5%
70	268.3	0.8%	0.5%	1.6%	1.5%

For the linear function, by taking the best of H1+RL algorithm and H2+RL algorithm, the average gap obtained over all the scenari is 0.51% and the worst gap reachs 2.5%.

Table 2. Iso-elastic demand, increasing seasonality over time

Production capacity	Exact method	H1	H1+RL	H2	H2+RL
50	218.5	2.5%	1.1%	0.9%	0.1%
60	224.1	4.4%	0.9%	3.3%	0.0%
70	227.0	0.9%	0.0%	4.6%	3.2%
80	228.3	1.1%	0.2%	0.2%	0.2%
90	230.9	1.9%	1.0%	1.1%	1.1%
100	231.8	0.9%	-0.4%	1.5%	1.5%
110	233.5	-0.1%	-0.1%	2.2%	1.5%

For the isoelastic function, by taking the best of H1+RL algorithm and H2+RL algorithm, the average gap obtained over all the scenari is 0.25% and the worst gap reachs 1% . The negative gaps in table 2 are due to the 0.5% optimality parameter used by the solver.

There is no clear dominant heuristic between the presented ones, it may depend on the structure of the problem, or it may depend on the initial data. Since their computation time stay low, it could be worthy to combine them to get the best solutions.

3 Perspectives and future research

A new efficient method aiming to solve coordinate production, inventory and pricing decisions is presented.

A direct extension of this work may be to work on the characterization based on the data of the initial sorting rules used in the first heuristic. The second heuristic has common features with the ”fix-and-relax” heuristic, it may be useful to implement it to compare the results.

Finally, the model allowed us to use two different demand functions, it may be challenging to implement a third one, such as the logit function [10], used to modelize market with competition.

Acknowledgment

This research was supported by the Industrial Chair Connected-Innovation.

References

1. Thomas, J. : Price-production decisions with deterministic demand. *Management science*, 16, 747–750, *INFORMS*
2. Smith, B. C., Leimkuhler, J. F. and Darrow, R. M.: Yield management at American airlines. *interfaces*, 22, 8–31, 1992, *INFORMS*
3. Bitran, G. and Caldentey, R.: An overview of pricing models for revenue management. *Manufacturing & Service Operations Management*, 5, 203–229, 2003, *INFORMS*
4. Haugen, K. K., Olstad, A. and Pettersen, B.: The profit maximizing capacitated lot-size (PCLSP) problem. *European Journal of Operational Research*, 176, 165–176, 2007, *Elsevier*
5. Ahmadi, M. and Shavandi, H.: Dynamic pricing in a production system with multiple demand classes. *Applied Mathematical Modelling*, 39, 2332–2344, 2015, *Elsevier*
6. Adida, E. and Perakis, G.: Dynamic pricing and inventory control: Uncertainty and competition. *Operations Research*, 58, 289–302, 2010, *INFORMS*
7. Fisher, M., Gallino, S. and Li, J.: Competition-based dynamic pricing in online retailing: A methodology validated with field experiments. *Management science*, 2017, *INFORMS*
8. Duran, M. A. and Grossmann, I. E.: An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical programming*, 36, 307–339, 1986, *Springer*
9. Bajwa, N., Sox, C. R. and Ishfaq, R.: Coordinating pricing and production decisions for multiple products. *Omega*, 64, 86–101, 2016, *Elsevier*
10. Huang, J., Leng, M. and Parlar, M.: Demand functions in decision modeling: A comprehensive survey and research directions. *Decision Sciences*, 44, 557–609, *Wiley Online Library*
11. Ouazene, Y., Yalaoui F., Kelly, R. and Idjeraoui, T.: Coordination and optimization of dynamic pricing and production decisions. *Computational Intelligence (SSCI)*, p1–6, *IEEE Symposium Series on*

Haralick Texture Features Selection For Ultrasound Image Segmentation By Multigene Genetic Programming

F.Z. Benabdallah[1] and L.Djerou[1]

LESIA Laboratory, University of Biskra, Algeria[1]
Fatima.benabdallah@hotmail.fr
ldjerou@yahoo.fr

Keywords :Haralick coefficients, Image segmentation, Multi gene genetic programming, Ultrasound image, Texture analysis, Region-based active contours.

1 Introduction

The segmentation of different organs and tissues in ultrasound images is one of the most challenging tasks in medical image processing [3]. The acquisition of these images is non-invasive, cheap, and does not require ionizing radiations compared to other medical imaging techniques. However, the ultrasound image segmentation is strongly influenced by the quality of data. There are characteristic artefacts which make the segmentation task complicated [2,3] such as shadows, speckle, attenuations, border attenuations, and it is sometimes difficult to obtain the desired results.

As described in [1], texture analysis is very efficient for ultrasound classification and segmentation. Texture is characterized by the spatial distribution of gray levels in a neighborhood. The segmentation of textured image consists in partitioning an image into homogeneous regions with respect to texture properties. The success of the segmentation depends mainly on the texture features selected to characterize the pixels of the image. Four principal texture feature families are identified [2] : statistical methods, geometric methods, model-based methods and frequency-based methods. Among the statistical features, Haralick coefficients of the co-occurrence matrix [4] which are commonly used in 2D texture analysis of medical images [2] and can be adaptable to 3D texture analysis. These features calculable both locally at the pixel level and over an entire region. However, they do not generally have the same discriminating power from one type of image to another. In this context, selecting the best Haralick coefficients (the weight of each Haralick coefficient) makes it possible to develop an efficient image segmentation model.

In this work, we use the multigene genetic programming (MGGP) for predicting the right texture features for the ultrasound image segmentation. Multigene Genetic programming (MGGP) technique [5] is a robust variant of genetic programming (GP), which effectively combines the model structure selection ability of standard GP with the parameter estimation power of classical regression by using a new characteristic called multi-gene.

By learning from data, the MGGP technique can find the best combination of the predictor Haralick features which are computed by the weighted output of each genes in the multigene program plus a bias term.

For the development of learning data, we adopt the same process presented in [3], which consists in extracting one single slice from the complete 3D ultrasound image and using it as a learning image. Then an expert segmentation C^* is performed on this learning image, composed of two ideal regions C_{in} and C_{out}. C_{in} will contain pixels from the object to be segmented while C_{out} will be composed of pixels belonging to a narrow band surrounding C_{in}, this segmentation type is illustrated in Figure 1.

Once the two regions (C_{in} and C_{out}) are determined, the 14 Haralick features (Table 1) are computed for each of their pixels. The Haralick features are determined using a neighborhood window (11x11) of pixels around every pixel of regions C_{in} and C_{out}.

The local Haralick texture features of pixels of this segmentation (C_{in} and C_{out}) are used to create the samples of a learning dataset, the pixels belonging to C_{in} have a label y=-1 and those belonging to C_{out} have a label y=+1. According to the expert segmentation (Figure 1), the learning dataset contains 5197 samples (pixels) with 1140 belong to C_{in} and 4057 belong to C_{out}.

To develop the proposed MGGP-based model for predicting the right texture features, we use GPTIPS2 [5] which is an open-source software platform for symbolic data mining in MATLAB.

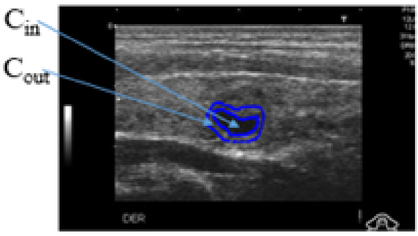


Figure 1. Example of expert segmentation

MGGP requires to define certain parameters, which are : iteration number, maximum number of genes allowed in an individual (Gmax), maximum tree depth (Dmax), crossover and mutation event probabilities and selection method, The functions used here are "+" and "-" for linear combination, and terminals are 14 Haralick texture features (see Table1).

The models are formed by randomly combining the elements from the functional set (+, -) and the terminal set (the 14 Haralick features). The optimal model is selected considering its simplicity as well as its performance on the learning data. In this context, two criteria are optimized simultaneously the complexity and the goodness-of-fit, by searching the so-called Pareto front (non-dominated solutions) set. The complexity of the model is defined as the sum of nodes of all sub-trees within a tree. Minimize this complexity is used to assess its simplicity of the model. The goodness-of-fit is determined by minimizing (1-R^2) :

$$R^2 = \frac{\sum_{i=1}^{n}(\hat{Y}_i - \bar{Y})^z}{\sum_{i=1}^{n}(Y_i - \bar{Y})^z} \tag{1}$$

Where : $\hat{y}_i$ and y_i are the measured and calculated output values for the i^{th} output, $\bar{y}$ respectively. is the average of the measured values for the output and n is the number of samples.

After 100 iterations on learning dataset with these parameters settings (Gmax=14, Dmax=3, tournament selection size= 20, Mutation probability=0.1, Crossover probability= 0.8 and Pareto tournament=0.3) which were determined experimentally observing convergence of the objectives functions over the generations, the following prediction model was selected :

0.237 x_2 - 0.1185 x_1 - 0.02162 x_4 + 4.396 x_5 - 0.09686 x_6 - 0.1937 x_9 + 1.786 x_{12} + 0.8928 x_{13} - 0.8928 x_{14} - 1.04

This predicting model is composed of 10 the best combination of the 10 weighted Haralick features plus a bias term. The Table1 shows the Haralick feature corresponding to each parameter x_i. According to this predictive model, we notice that, the feature 5 (x_5 : Inverse Difference Moment) is the most important because it has the largest weight in the model.

Table 1. Haralick texture features

x_1 : Angular Second Moment (Energy)	x_6 : Sum Avergae	x_{10} : Difference Variance
x_2 : Contrast	x_7 : Sum Variance	x_{11} : Difference Entropy
x_3 : Correlation	x_8 : Sum Entropy	x_{12} : Information Measure of Correlation I
x_4 : Variance	x_9 : Entropy	x_{13} : Information Measure of Correlation II
x_5 : Inverse Difference Moment		x_{14} : Maximal Correlation Coefficient

Once the weighted Haralick features are determined by MGGP, we can segment the series of images with properties of textures similar to that of the learning image. In this case, we have used a region-based active contour model presented in [6] whose evolution from the curve C in the image, is based on the energy functional E of the model of Chan and Vese, but instead of relying on the gray levels present in the image, it will use the selected weighted Haralick features for each pixel of the image, as it is presented in the following equation :

$$E(C,\bar{C_j^{in}},\bar{C_j^{out}}) = (\int_{C_{in}} \sum_{j=1}^{m} w_j |X_j(P) - \bar{C_j^{in}}|^z dp * g^{in(P)}) + (\int_{C_{out}} \sum_{j=1}^{m} w_j |X_j(P) - \bar{C_j^{out}}|^z dp * g^{out(P)}) \tag{2}$$

Where $\bar{C_j^{in}}$ and $\bar{C_j^{out}}$ are the mean values of the j^{th} Haralick feature for the regions C_{in} and C_{out} respectively.X_j(P) is the value of j^{th} Haralick feature for the pixel P. w_j is the weight associated with the j^{th} Haralick feature. g^{out} and g^{in} are the membership functions of the pixel P in C_{out} and C_{in} respectively. The following figure (Figure 2) presents the segmentation results of three thyroid ultrasound images, belong to a complete series of thyroid ultrasound images, each one has properties of textures similar to that of the learning image (a). These results encourage us to continue in improving this research.

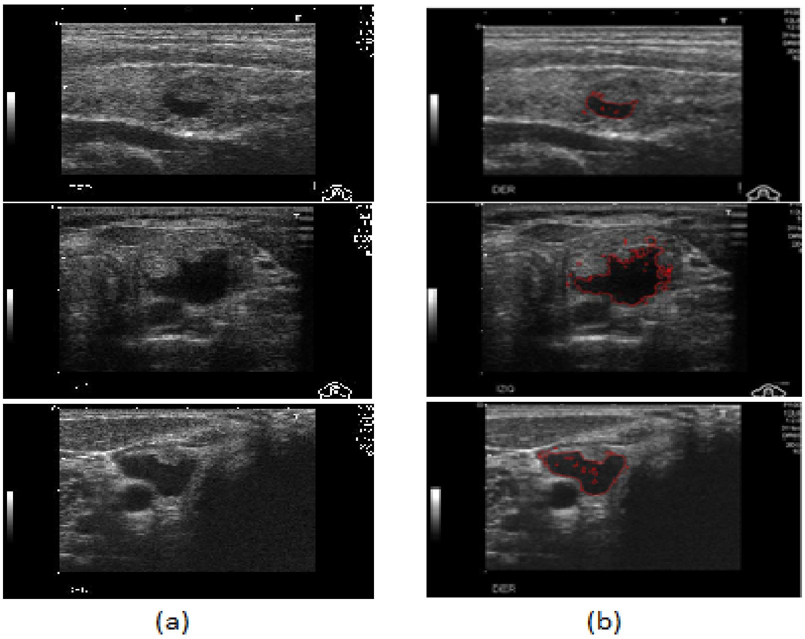


Figure 2. Thyroid ultrasound images segmentation : (a) Slices of 3D thyroid ultrasound original image, (b) Segmentation result.

Références

1. Noble, J. A., & Boukerroui, D. (2006). Ultrasound image segmentation : a survey. IEEE Transactions on medical imaging, 25(8), 987-1010.
2. Sciolla, B., Cowell, L., Dambry, T., Guibert, B., & Delachartre, P. (2017). Segmentation of skin tumors in high-frequency 3-D ultrasound images. Ultrasound in Medicine and Biology, 43(1), 227-238.
3. Olivier, J., & Paulhac, L. (2011). 3D ultrasound image segmentation : Interactive texture-based approaches. In Medical Imaging. InTech.
4. Haralick, R. M., & Shanmugam, K. (1973). Textural features for image classification. IEEE Transactions on systems, man, and cybernetics, (6), 610-621.
5. Searson, D. P. (2015). GPTIPS 2 : an open-source software platform for symbolic data mining. In Handbook of genetic programming applications (pp. 551-573). Springer, Cham.
6. Dirami, A., Hammouche, K., Diaf, M., & Siarry, P. (2013). Fast multilevel thresholding for image segmentation through a multiphase level set method. Signal Processing, 93(1), 139-153.

An Energy-Efficient Permutation Flowshop Scheduling

M. Fatih Tasgetiren[1], Hande Öztop[2], Quan-Ke Pan[3], M. Arslan Örnek[4], Talya Temizceri[5],

[1]Department of International Logistics, Yasar University, İzmir, Turkey
fatih.tasgetiren@yasar.edu.tr
[2, 4, 5]Department of Industrial Engineering, Yasar University, İzmir, Turkey
hande.oztop, arslan.ornek, talya.temizceri}@yasar.edu.tr
[3]State Key Laboratory, Huazhong University of Science and Technology, Wuhan, P.R. China
panquanke@hust.edu.cn

Abstract. Permutation flow shop scheduling problem is a well-known problem in the literature. Even though many multi-objective permutation flowshop scheduling problems are presented in the literature, energy consumption consideration in scheduling is still very seldom in the literature. In this paper, we consider a bi-objective permutation flowshop scheduling problem and propose a bi-objective mixed integer linear programming model for the objectives of minimizing the total energy consumption and the makespan in order to see the trade-off between them. We employ the augmented-epsilon constraint method for generating the Pareto optimal solution sets for small sized instances. For larger instances, we use the augmented epsilon-constraint method with a time limit on CPLEX for approximating the Pareto solution sets. As a heuristic method, we employ a very recent variable block insertion heuristic algorithm from the literature. First, we show the performance of the proposed algorithm on small sized problems; then, it is shown that the proposed algorithm is very effective for solving larger instances when compared to the time-limited CPLEX.

Keywords: Permutation flowshop scheduling, energy efficient scheduling, bi-objective optimization, variable block insertion algorithm, heuristic optimization.

1 Introduction

In the permutation flowshop scheduling problem (PFSP), a set of n jobs is to be processed on m machines. Each job consists of m operations, and each operation is to be performed by one of the m machines. More formally, given an arbitrary solution, i.e., a permutation of jobs, $\sigma = \{\sigma_1, \sigma_2, \dots, \sigma_n\}$, job σ_i is the job at the i^{th} position of solution σ. Each job has a processing time denoted as $p_{\sigma_{i,k}}$. Ready times are zero and same permutation is used on all machines. Let $C(i,k)$ be the completion time of job σ_i on machine k at position i. Completion times of jobs at each machine are computed as follows:

$$C(1,1) = p_{\sigma_{1,1}} \tag{1}$$

$$C(i,1) = C(i-1,1) + p_{\sigma_{i,1}} \qquad \forall i = 2, \dots, n \tag{2}$$

$$C(1,k) = C(1,k-1) + p_{\sigma_{1,k}} \qquad \forall k = 2,\dots,m \tag{3}$$

$$C(i,k) = \max\{C(i-1,k), C(i,k-1)\} + p_{\sigma_{i,k}} \quad \forall i = 2,\dots,n; \forall k = 2,\dots,m \tag{4}$$

The makespan, denoted as $C_{max}(\sigma)$, is the completion time of the last job on the last machine. It is simply denoted as $C(n,m)$. The aim is to find a permutation of jobs minimizing the makespan.

In the PFSP literature, tardiness and flow time based performance measures have been generally discussed in order to measure production efficiency and customer satisfaction. However, energy efficiency in production scheduling has been considered rarely. Recently, the energy consumption has become a key concern for manufacturing sector because of the negative environmental impacts such as gas emissions (CO_2) and global warming. As the manufacturing facilities consume high energy, they are forced to reduce their energy consumption by developing more energy efficient scheduling systems [1].

An energy efficient scheduling framework was outlined in [2]. As a pioneering study, it was concluded that once machines are turned off at idle times, considerable amount of energy can be saved [3]. Later on, this turn off strategy was employed in the single machine-scheduling problem that minimizes total energy consumption and total tardiness in [4]. Similarly, turn off strategy was employed for the flexible flowshop problem in [5]. Due to the inefficiency of turn off strategy, a speed scaling strategy was first developed for the energy-efficient FSP [1]. They considered operation speed as an independent variable that can be adjusted to improve energy efficiency. Later, a MIP formulation was given in [6] for the PFSP considering makespan as an objective and peak power consumption as a constraint. In addition, the speed scaling strategy was used for the PFSP that minimizes the total carbon emissions and the makespan [7]. Recently, a multi-objective genetic algorithm was presented in Zhang & Chiong [8] in order to minimize the total weighted tardiness and total energy consumption in a job shop scheduling. Variable speed levels were also studied for the two machines PFSP with sequence-dependent setup times and some lower bounds as well as a heuristic were proposed [9]. In addition, an energy efficient permutation flowshop scheduling using backtracking algorithm was developed in [10] whereas in [11], an energy efficient evolutionary algorithm was proposed for single machine scheduling with sequence-dependent setup times.

In this paper, a bi-objective mixed integer linear programming model (MILP) is developed for the PFSP using speed-scaling strategy in order to investigate the trade of between C_{max} and the total energy consumption (TEC). In addition, we propose an energy-efficient variable block insertion heuristic (VBIH) algorithm employing the speed scaling strategy similar to those proposed in [7] and [9]. In fact, we were inspired by these two notable papers. In these two works, they employed a matrix representation for speed scaling strategy. In other words, for each machine, a different speed scaling strategy is employed. However, we employ a simple job-based speed scaling strategy where the same speed strategy is used on all machines.

The remainder of this paper is organized as follows. Section 2 gives the problem formulation for the bi-objective MILP model. Section 3 explains the bi-objective energy-efficient VBIH (EE_VBIH) algorithm. In Section 4 presents the computational results and the conclusion is given with future research directions in Section 5.

2 Problem formulation

In this section, we formulate the problem as a bi-objective PFSP examining a tradeoff between C_{max} and TEC. Notations used in the mathematical model are given in Table 1.

Table 1. Notations

Sets and Indexes	
L	Set of speed levels
j	Index for machines $(1 \leq j \leq M)$
i and k	Index for jobs $(1 \leq i, k \leq N)$
Parameters	
p_{ij}	Processing time of job i on machine j
v_l	Speed factor of speed l
λ_l	Processing conversion factor for speed l
φ_j	Conversion factor for idle time on machine j
τ_j	Power of machine j (kW)
D	A very large number
Decision Variables	
y_{ijl}	1 if job i is processed at speed l on machine j; 0,otherwise
x_{ik}	1 if job i precedes job k; 0 otherwise $(i < k)$
C_{ij}	Completion time of job i on machine j
θ_j	Idle time on machine j
C_{max}	Maximum completion time (makespan)
TEC	Total energy consumption (kWh)

The bi-objective MILP formulation is given below:

$$Min\ C_{max}, TEC \qquad (1)$$

$$C_{i1} \geq \sum_{l \in L} \frac{P_{i1} y_{i1l}}{v_l} \qquad (1 \leq i \leq N) \qquad (2)$$

$$C_{ij} - C_{i,j-1} \geq \sum_{l \in L} \frac{P_{ij} y_{ijl}}{v_l} \qquad (2 \leq j \leq M; 1 \leq i \leq N) \qquad (3)$$

$$C_{ij} - C_{kj} + Dx_{i,k} \geq \sum_{l \in L} \frac{P_{ij} y_{ijl}}{v_l} \qquad (1 \leq j \leq M; 1 \leq i < k \leq N) \qquad (4)$$

$$C_{ij} - C_{kj} + Dx_{ik} \leq D - \sum_{l \in L} \frac{P_{kj} y_{kjl}}{v_l} \qquad (1 \leq j \leq M; 1 \leq i < k \leq N) \qquad (5)$$

$$C_{max} \geq C_{iM} \qquad (1 \leq i \leq N) \qquad (6)$$

$$\sum_{l \in L} y_{ijl} = 1; \qquad (1 \leq i \leq N; 1 \leq j \leq M) \qquad (7)$$

$$y_{ijl} = y_{i,j+1,l} \qquad (1 \leq i \leq N; 1 \leq j < M; 1 \leq l \leq \mathrm{L}) \qquad (8)$$

$$\theta_j = C_{max} - \sum_{i=1}^{N} \sum_{l \in L} \frac{P_{ij} y_{ijl}}{v_l} \qquad (1 \leq j \leq M) \qquad (9)$$

$$TEC = \sum_{i=1}^{N} \sum_{j=1}^{M} \sum_{l \in L} \frac{P_{ij} \tau_j \lambda_l}{60 v_l} y_{ijl} + \sum_{j=1}^{M} \frac{\varphi_j \theta_j \tau_j}{60} \qquad (10)$$

Objective function (1) minimizes C_{max} and TEC. Constraint (2) ensures that the completion time of each job on machine 1 is greater than or equal to its processing time on machine 1. Constraint (3) states that a job can be started only after its preceding operation has been completed. Constraints (4) and (5) guarantee that job i either precedes job k or vice versa in the sequence, but not both. Constraint (6) computes the makespan, which is the maximum completion time of all jobs on the last machine. Constraint (7) and (8) ensure that exactly one speed level is selected for each job and the same speed level is employed on every machine. Constraint (9) calculates the idle time of each machine. Constraint (10) computes the total energy consumption as proposed by [9].

As mentioned above, our problem is a multi-objective problem, we present the dominance relation concepts [12] that will be used when solving the PFSP:

- **Definition 1: Dominance relation**. A solution x_i dominates another solution x_j if the two following conditions are satisfied (denoted as $x_i \prec x_j$):
    - ○ $\forall p \in 1, .., P; f_p(x_i) \leq f_p(x_j)$
    - ○ $\exists p \in 1, .., P; f_p(x_i) < f_p(x_j)$

  A solution x_i weakly dominates another solution x_j (denoted as $x_i \preccurlyeq x_j$) if :
    - ○ $\forall p \in 1, .., P; f_p(x_i) \leq f_p(x_j)$

  A solution x_i is indifferent to another solution x_j (denoted as $x_i \backsim x_j$) if :
    - ○ $\forall p \in 1, .., P; f_p(x_i) \nleq f_p(x_j) \wedge f_p(x_j) \nleq f_p(x_i)$
- **Definition 2: Non-dominated set**: Amongst a set of solutions X, the non-dominated set of solutions are the elements of the set X^* non-dominated by any element of the set X.
- **Definition 3: Pareto-optimal set**: The non-dominated set of the entire feasible search space S is called the Pareto-optimal solutions.

There are common solution methods for multi-objective problems such as sequential optimization, goal programming, weighting method and ε-constraint method. In this study, we prefer to use augmented ε-constraint method, as it generates only Pareto-optimal solutions [13]. In Pareto-optimal solutions, any objective function cannot be improved without worsening another objective function. In augmented ε-constraint method, one of the objective functions is optimized, while other objective functions are defined by constraints. Dissimilar to the standard ε-constraint method, slack/surplus variables are included in these objective function constraints by converting them to equalities. These variables are also defined as second term in the objective function to ensure the Pareto-optimality.

3 Energy-efficient VBIH algorithm

Recently, block move-based search algorithms are presented for scheduling problems in literature [14-19]. The VBIH algorithm in this paper simply removes a block b of jobs from the current solution; then it makes a number $n - b + 1$ of block insertion moves on the partial solution denoted as $bMove()$ procedure. Then, the best one from the $bMove()$ procedure is retained in order to undergo a local search procedure. If the new solution obtained after the local search is better than the current solution, it replaces the current solution. Otherwise, a simple simulated annealing-type of acceptance criterion is used with a constant temperature, which is suggested by [20], as follows: $T = \frac{\sum_{i=1}^{n}\sum_{k=1}^{m} p_{ik}}{10nm} \times \tau P$, where τP is a parameter to be adjusted. Initially, the block size is fixed to $b = 1$. As long as it improves, it retains the same block size ($i.e., b = b$). Otherwise, it is increased by one ($i.e., b = b + 1$). the $bMove()$ procedure is carried out until the block size reaches at the maximum block size (i.e., $b \leq b_{max}$). The outline of the VBIH algorithm is given in Fig. 1.

```
 $\sigma = NEH$
 $\sigma_{best} = \sigma$
 $while\ (NotTermination)\ do$
$b = 1$
   $do\{$
        $\sigma_1 = bMove(\sigma,)$
        $\sigma_1 = LocalSearch(\sigma_1)$
        $if\ (f(\sigma_1) \leq f(\sigma))\ then\ do\{$
             $\sigma = \sigma_1$
             $if\ f(\sigma_1) < f(\sigma_{best})\ then\ do\{$
                  $\sigma_{best} = \sigma_1$
                  $b = b$
        $else\{$
            $b = b + 1$
            $if\ (r < exp\{-(f(\sigma) - f(\sigma_1))/T\})$
                $\sigma = \sigma_1$
        $\}endif$
    $\}while(b \leq b_{max})$
  $\}endwhile$
  $return\ \sigma_{best}\ and\ f(\sigma_{best})$
  $endprocedure$
```

Fig. 1 Variable block insertion heuristic

In this study, a job-based speed scaling strategy is proposed for the energy-efficient VBIH algorithm. To handle the speed scaling strategy, a multi-chromosome structure is used. It is composed of a permutation of n jobs (σ) and a speed vector of three levels (v). Three speed levels correspond to fast, normal and slow speed levels, respectively. The solution representation is given in Fig. 2.

$x\binom{\sigma}{v}$	σ	5	2	1	4	3	$\dots$	n
	v	3	1	2	1	2	$\dots$	3

Fig. 2. Solution Representation.

In Figure 2, a solution/individual $x\binom{\sigma}{v}$ indicates that job $\sigma_1 = 5$ has a slow speed level, $(i.e.,$ $v_3 = 3)$, job $\sigma_2 = 2$ has a fast speed level, $(i.e., v_1 = 1)$; and so on.

3.1 Initial Population

For the initial population with size NP, the following procedure is used: A solution is constructed by the NEH heuristic [21]. This solution is taken as an initial solution for the VBIH algorithm with makespan minimization only. Ten percent of the total CPU time budget is devoted to the VBIH algorithm in order to obtain a good starting point for the EE_VBIH algorithm. Once the best solution σ_{best} is found by the VBIH algorithm, the first three solutions in population are obtained by assigning fast, normal or slow speed levels to each job in the best solution σ_{best}. The rest of the population is obtained by assigning random speed levels between 1 and 3 to each job in the best solution σ_{best}. The archive set Ω is initially empty and it is updated.

3.2 Block insertion procedure

The $bMove()$ procedure is a core function in the EE_VBIH algorithm. The procedure randomly removes a block b of jobs with their speed from the current solution. Then, block is denoted by x^b whereas the partial solution after removal will be denoted by $x^p = (J - x^b)$. First, speed levels in x^b are randomly changed between 1and 3. Then; similar to the one presented in [22], the EE_VBIH algorithm applies an additional local search to partial solution x^p before carrying out a block insertion. Then; the $bMove()$ procedure carries out $n - b + 1$ block insertion moves. In other words, block x^b is inserted in all possible positions in the partial solution x^p. It should be noted that dominance rule $(\prec)$ explained before is used when two solutions and/or partial solutions are compared

In order to explain the $bMove()$ procedure, following example would be useful. Suppose that we have a current solution $x\binom{\sigma}{v} = \begin{Bmatrix}3, 1, 4, 2, 5\\2, 1, 3, 1, 2\end{Bmatrix}$ with block size $b = 2$. A block is removed and two partial solutions are obtained as follows: $x^b\binom{\sigma}{v} = \binom{1, 4}{1, 3}$ and $x^p\binom{\sigma}{v} = \binom{3, 2, 5}{2, 1, 2}$. First, speed levels of x^b are randomly changed to, say, $x^b\binom{\sigma}{v} = \binom{1, 4}{3, 2}$. Then; an insertion local search is applied to the partial solution x^p in a way that each job and speed pair is removed from x^p and inserted into all positions without the position it is removed. The best non-dominated partial solution is retained. Suppose that the best one is $x^p\begin{pmatrix}\sigma\\v\end{pmatrix}=\begin{pmatrix}5, 2, 3\\2, 1, 2\end{pmatrix}$. Finally, the block x^b is inserted into all positions in x^p as follows: $x\begin{pmatrix}\sigma\\v\end{pmatrix}=\begin{pmatrix}1, 4, 5, 2, 3\\3, 2, 2, 1, 2\end{pmatrix}$, $x\begin{pmatrix}\sigma\\v\end{pmatrix}=\begin{pmatrix}5, 1, 4, 2, 3\\2, 3, 2, 1, 2\end{pmatrix}$, $x\begin{pmatrix}\sigma\\v\end{pmatrix}=\begin{pmatrix}5, 2, 1, 4\ 3\\2, 1, 3, 2, 2\end{pmatrix}$, and $x\begin{pmatrix}\sigma\\v\end{pmatrix}=\begin{pmatrix}5, 2, 3, 1, 4\\2, 1, 2, 3, 2\end{pmatrix}$. Among these four solutions, the non-dominated one will be selected and the archive set Ω is updated.

3.3 Energy-efficient insertion local search

Regarding the local search algorithm, we employ a very effective first-improvement insertion neighborhood structure for each individual i in the population. Similar to the $bMove()$ procedure, each job and speed level is removed from the current solution and inserted into all positions of the current solution. The non-dominated solution is retained and the archive set Ω is updated. The insertion local search has a size of $(n-1)^2$. As an example, we consider the solution above $x\begin{pmatrix}\sigma\\v\end{pmatrix}=\begin{Bmatrix}3, 1, 4, 2, 5\\2, 1, 3, 1, 2\end{Bmatrix}$. The first job and its speed level, $\begin{Bmatrix}3\\2,\end{Bmatrix}$ are removed from the current solution x. Its speed level is randomly changed to another speed level, say, $\begin{Bmatrix}3\\1,\end{Bmatrix}$. Then; they are inserted into all positions in the solution x as follows: $x\begin{pmatrix}\sigma\\v\end{pmatrix}=\begin{Bmatrix}3, 1, 4, 2, 5\\1, 1, 3, 1, 2\end{Bmatrix}$, $x\begin{pmatrix}\sigma\\v\end{pmatrix}=\begin{Bmatrix}1,3, 4, 2, 5\\1,1,3,1, 2\end{Bmatrix}$, $x\begin{pmatrix}\sigma\\v\end{pmatrix}=\begin{Bmatrix}1,4, 3, 2, 5\\1,3,1,1, 2\end{Bmatrix}$, $x\begin{pmatrix}\sigma\\v\end{pmatrix}=\begin{Bmatrix}1,4, 2, 3, 5\\1,3,1,1, 2\end{Bmatrix}$, $x\begin{pmatrix}\sigma\\v\end{pmatrix}=\begin{Bmatrix}1,4, 2, 5, 3\\1,3,1,2, 1\end{Bmatrix}$. Among these five solutions, the non-dominated one will be selected and the archive set Ω is updated. This is repeated for the next pair of job and its speed level until the last job and its speed level are inserted into all positions.

3.4 Energy-efficient uniform crossover and mutation

In order to obtain more energy-efficient schedules, a local search algorithm based on uniform crossover operator by considering only speed levels is proposed in this paper. Note that with the same permutation, any change in speed levels leads to a different solution in terms of $C_{max}\ and\ TEC$. For this reason, after having applied the VBIH algorithm to each individual in the population, the same permutation is kept for each individual in the population and a uniform crossover on speed levels is carried out as follows. For each individual x_i in the population, we select another individual from population randomly, say x_k, new solution is obtained in a way that taking the speed level is either taken from x_i or x_k with a crossover probability $pC[i]$. The uniform crossover is carried out as follows:

$$x_{new}\begin{pmatrix}\sigma\\v\end{pmatrix}=\begin{cases}x_i(v_{ij}) & if\ r_{ij}\le pC[i]\\ x_k(v_{kj}) & otherwise\end{cases}\qquad \forall j\in 1,..,n$$

where r_{ij} is a uniform random number in $U(0,1)$ and $pC[i]$ is the crossover probability, which is drawn from unit normal distribution $N(0.5,0.1)$ for each individual x_i in the population. If x_{new} dominates x_i (i.e., $x_{new}\prec x$), x_i is replaced by x_{new} and the archive set Ω is updated. This is repeated for all individuals in the population.

After having carried out uniform crossover for all individuals in the population, we mutate the population by lowering the speed levels with a small mutation probability as follows:

$$x_i(\sigma_{ij},v_{ij})=\begin{cases}x_i(v_{ij}=1+rand()\%2) & if\ r_{ij}\le pM[i]\\ x_i(v_{ij}) & otherwise\end{cases}\qquad \forall j\in 1,..,n;\ \forall i\in 1,..,NP$$

where r_{ij} is a uniform random number in $U(0,1)$ and $pM[i]$ is the mutation probability, which is drawn from unit normal distribution $N(0.05,0.01)$ for each individual x_i in the population.

3.5 The archive set

An archive set Ω is used to store the non-dominated solutions during the optimization process. This archive set should be updated with non-dominated solutions in order to approximate the

Pareto-optimal solutions. When a new non-dominated solution is obtained, it should be added to the archive set Ω and any member dominated by the new non-dominated solution should be removed.

3.5.1 Update archive set

In order to update the archive set Ω, Pan et.al. [23] proposed an effective method for updating the archive set as follows: The non-dominated solutions in Ω are stored in increasing order of their first objective function values. Then, their second objective values will be in decreasing order. The procedure for updating the archive set Ω can be summarized as follows:

Step 1. Archive size is $m = |\Omega|$ and $\Omega = \{a_1, a_2, .., a_m\}$. Initially, Ω is empty and the first non-dominated solution x will be added to the first position in Ω. Let $k = 1$.

Step 2. Find a most suitable position pos for the next individual x in the archive set Ω by the following procedure:

```
do{
   $j = \lfloor (k+m)/2 \rfloor$
   $if\ \left(f_1(x) = f_1(a_j)\right)\ then\ j = j$
   $elseif\ \left(f_1(x) < f_1(a_j)\right)\ then\ m = j - 1$
   $else\ \ k = j + 1$
$while(k \leq m)$
```

Step 3. When comparing $f_1(x)$ with $f_1(a_j)$, following cases may occur:

$Case$ 1. $if\ \left(f_1(x) = f_1(a_j)\right)\ and\ \ if\ \left(f_2(x) < f_2(a_j)\right) then\ pos = j$

$Case$ 2. $if\ \left(f_1(x) < f_1(a_j)\right)$

$if\ \ j = 1\ \ \ then\ pos = j\ and\ m = m + 1$

$if\ \ j > 1\ and\ \ if\ \left(f_2(x) < f_2(a_{j-1})\right) then\ pos = j\ and\ m = m + 1$

$Case$ 3. $if\ \left(f_1(x) > f_1(a_j)\right)\ and\ \ if\ \left(f_2(x) < f_2(a_j)\right) then\ pos = j + 1\ and\ m = m + 1$

If any of cases above is satisfied, solution x is added to position pos, but all solutions dominated by x in Ω should be removed. The following procedure removes the dominated solutions from Ω:

Step 1. $If\ (pos = m)\ then\ go\ to\ Step$ 4
Step 2. $Let\ pos = pos + 1.\ \ If\ f_2(a_{pos}) \geq f_2(x)\ then\ remove\ a_{pos};\ otherwise\ go\ to\ Step$ 4
Step 3. $if(pos < m)\ then\ go\ to\ Step$ 2
Step 4. $\Omega = report\ non - dominated\ solutions$

3.5.2 Crowding distance

For a solution in Ω, the crowding distance is the sum of the normalized distance between its previous and next neighbors for each objective function value. The extreme solutions have the crowding distance set to infinity. It is clear that the larger the crowding distance, the sparser the nearby solutions. Based on the storage structure of Ω, the crowding distance of a non-dominated solution a_j is given as follows:

$$cD_j = \begin{cases} \infty & if\ (j = 1\ or\ j = s) \\ \frac{f_1(a_{j+1}) + f_1(a_{j-1})}{f_1(a_s) - f_1(a_1)} + \frac{f_2(a_{j-1}) + f_2(a_{j+1})}{f_2(a_1) - f_1(a_s)} & otherwise \end{cases}$$

5 Computational Results

To test the performance of the algorithms, extensive experimental evaluations are carried out on the benchmark suite of Taillard [24]. In this paper, we only employ the first 60 instances from 20 jobs and 5 machines to 50 jobs and 20 machines (20x5, 20x10, 20x20, 50x5, 50x10 and 50x20). In addition, due to the computational difficulty of bi-objective problem, we generate 30 small-sized instances with 5 jobs and 5 machines, 5 jobs and 10 machines, 5 jobs and 20 machines by truncating 20x5, 20x10 and 20x20 problems. Population size is taken as NP=100. For the VBIH with makespan minimization only, the maximum block size is taken as $b_{max} = 5$; and temperature for acceptance criterion are taken as $\tau P = 0.4$. The speed and conversion parameters are taken as $v_l = \{1.2, 1, 0.8\}$, and $\lambda_l = \{1.5, 1, 0.6\}$ for the fast, normal and slow speed levels, respectively. The power of machines are the same (60 kW) and the conversion factor for idle time is 0.05.

All instances are solved with the augmented ε-constraint method using IBM ILOG CPLEX 12.6.3 on a Core i7, 2.60 GHz, 8 GB RAM computer. We minimize C_{max} by defining TEC with a constraint. Initially, we obtain the ranges of each objective from payoff tables using lexicographic optimization. Afterwards, we solve the single-objective model repetitively by reducing the constraint on TEC with a specific ε level. We obtain very close approximations for the Pareto-optimal frontiers of instances with 5 jobs (5x5, 5x10 and 5x20) choosing an ε level as 10^{-3}. These finite numbers of Pareto-optimal solutions are named as Pareto-optimal solution set (P). Due to the exponentially increasing solution times, we find non-dominated solution sets for larger instances using a relatively higher ε level, which is calculated by dividing the range of TEC objective function to 20 equal grids. We set 3 minutes time limit in each iteration for these large instances.

The EE_VBIH algorithm is coded in C++ programming language on Microsoft Visual Studio 2013. Five replications are carried out for each instance. In each replication, the algorithm is run for $10nm$ milliseconds for small instances and $30nm$ milliseconds for larger instances, where n is the number of jobs and m is the number of machines. It is important to note that we initially set the archive size to $m = 5 \times NP$ in each replication. After five replications, we keep only non-dominated solutions in Ω because a solution in a replication can dominate a solution in another replication. Due to the real values of objective functions, we generate as many as non-dominated solutions after 5 replications. However, we compute the crowding distances of all these solutions and we only report the most crowded solutions up to $m = 100$.

As we find very close approximations to Pareto-optimal frontiers for instances with five jobs, we use below performance measures to evaluate the solution quality of the EE_VBIH algorithm. I refers to the non-dominated solution set of the EE_VBIH algorithm.

- Ratio of the Pareto-optimal solutions found: $R_p = |I \cap P|/|P|$
- Inverted Generational Distance [25]: $IGD = \sum_{v \in P} d(v, I)/|P|$, where $d(v, I)$ indicates the minimum Euclidean distance between v and the solution in I. The low IGD value means that set I is very close to set P.
- Distribution Spacing [26]; $DS_I = \frac{\left[\frac{1}{|I|}\sum_{i \in I}(d_i - \bar{d})^2\right]^{1/2}}{\bar{d}}$, where $\bar{d} = \frac{\sum_{i \in I} d_i}{|I|}$, d_i is the minimum Euclidean distance between solution i and its closest neighbour in I. Low spacing value shows that the solutions in I are uniformly distributed.

Table 2. Comparison of EE_VBIH and CPLEX on small-sized instances

Instance Set	**Rp**	**IGD**	**DS_I**
5x5	0.9820	0.00003	0.7000
5x10	0.8170	0.00023	0.8346
5x20	0.6360	0.00055	0.8902

Average	**0.8117**	**0.00027**	**0.8083**

Table 2 reports the average results of R_p, IGD and DS measures for each small-sized instance set, where there are 10 instances in each set. As shown in the table, the EE_VBIH algorithm finds approximately 82% of the Pareto-optimal solutions. Especially, for eight instances, all Pareto-optimal solutions are found by EE_VBIH algorithm. Furthermore, average IGD value of EE_VBIH algorithm is very low (0.00027), which indicates that the EE_VBIH provides very close approximations to the Pareto-optimal solution sets. In terms of distribution spacing, we can say that solutions in *I* are evenly distributed due to the low DS value.

For large instances, non-dominated solution sets of EE_VBIH algorithm (*I*) and time-limited CPLEX (*M*) are compared with each other in terms of the below performance metrics and the aforementioned DS metric (averaged over ten instances).

- Cardinality: the number of non-dominated solutions found.
- Coverage of Two Sets (*C*) [27]

  $C_{IM} = |m \in M;\ \exists i \in I: i \preccurlyeq m\}|/|M|$

where C_{IM} equals 1 if some solutions of *I* weakly dominate all solutions of *M*.

Table 3. Comparison of EE_VBIH and CPLEX on large instances

Instance Set	|M|	|I|	C_{MI}	C_{IM}	DS_M	DS_I
20x5	18.300	100.000	0.057	0.651	0.181	2.019
20x10	16.300	93.800	0.013	0.786	0.212	2.481
20x20	16.900	83.700	0.020	0.808	0.301	2.289
50x5	14.400	100.000	0.000	0.866	0.415	4.428
50x10	9.000	80.700	0.001	0.957	0.737	4.271
50x20	6.900	67.100	0.000	1.000	0.941	4.115
Average	**13.633**	**87.550**	**0.015**	**0.845**	**0.464**	**3.267**

Table 3 reports the average results for *M* and *I* for each large instance set, where there are 10 instances in each set. As shown in the table, EE_VBIH generates approximately seven times as many non-dominated solutions in very reasonable computation times. Furthermore, EE_VBIH performs much better than the time-limited CPLEX in terms of coverage metric, since 85% of the solutions of *M* are weakly dominated by some solutions of *I*. Particularly, some solutions of *I* weakly dominate all solutions of the *M* , in 17 of the instances. In terms of distribution spacing, solutions in *M* are distributed more uniformly than the solutions in *I*, as a fixed ε level is employed through the augmented ε-constraint method in time-limited CPLEX.

6 Conclusion

This paper presents an energy-efficient PFSP with minimization of makespan and total energy consumption. We proposed a simple job-based speed scaling strategy, which is the same for all machines in the problem. We develop a multi-objective MILP model and a heuristic algorithm. Small-sized instances are generated from Taillard's benchmarks to find Pareto optimal solution sets. First, the MILP model is run for these toy instances and we obtained Pareto optimal solution sets. For the larger instances, we employed time limited CPLEX to find 20 solutions for each instance. For toy instances, the EE_VBIH algorithm was able to find approximately 82% of the Pareto-optimal solutions. Especially, for eight instances, all Pareto-optimal solutions are found by EE_VBIH algorithm. For larger instances, EE_VBIH generates approximately eight times as many non-dominated solutions in very reasonable computation times. Furthermore, EE_VBIH performs much better than the time-limited CPLEX in terms of coverage metric, since 85% of the solutions of time-limited CPLEX are dominated by the EE_VBIH algorithm. In particular, EE_VBIH weakly dominates all solutions of time-limited CPLEX in 17 out of 60 instances.

For further research, the matrix representation for speed scaling strategy can be easily adapted by modifying the MILP model and EE_VBIH algorithm. Some multi-objective metaheuristic algorithms can be employed and different performance measures such as total tardiness and total flow time criteria can be another research direction.

References

[1]. K. Fang, N. Uhan, F. Zhao, J.W. Sutherland. 2011. A New Approach to Scheduling in Manufacturing for Power Consumption and Carbon Footprint Reduction. Journal of Manufacturing Systems. 30(4): 234–240.

[2]. C. Gahm, F. Denz, M. Dirr, A. Tuma. 2016. Energy-Efficient Scheduling in Manufacturing Companies: A Review and Research Framework. European Journal of Operational Research. 248: 744-757.

[3]. G. Mouzon, M.B. Yildirim, J. Twomey. 2007. Operational Methods for the Minimization of Energy Consumption of Manufacturing Equipment. International Journal of Production Research. 45 (18-19): 4247–4271

[4]. G. Mouzon, M.B. Yildirim. 2008. A Framework to Minimize Total Energy Consumption and Total Tardiness on a Single Machine. International Journal of Sustainable Engineering. 1(2): 105–16.

[5]. M. Dai, D. Tang, A. Giret, M.A. Salido, W. Li. 2013. Energy-Efficient Scheduling for a Flexible Flowshop Using An Improved Genetic-Simulated Annealing Algorithm. Robotics and Computer Integrated Manufacturing. 29: 418–429

[6]. K. Fang, N.A. Uhan, F. Zhao, J.W. Sutherland. 2013. Flowshop Scheduling with Peak Power Consumption Constraints. Annals of Operations Research. 206: 115–145.

[7]. J-Y. Ding, S. Song, C. Wu. 2016. Carbon-Efficient Scheduling of Flowshops by Multi-Objective Optimization. European Journal of Operational Research. 248: 758-771.

[8]. R. Zhang, R. Chiong. 2016. Solving the Energy-Efficient Job Shop Scheduling Problem: A Multi-Objective Genetic Algorithm with Enhanced Local Search for Minimizing the Total Weighted Tardiness and Total Energy Consumption. Journal of Cleaner Production. 112: 3361-3375.

[9]. S.A. Mansouri, E. Aktas, U. Besikci. 2016. Green Scheduling of a Two-Machine Flowshop: Trade-off between Makespan and Energy Consumption. European Journal of Operational Research. 248: 772-788.

[10]. C. Lu, L. Gao, X. Li, QK Pan, Q. Wang. 2017. Energy-Efficient Permutation Flow Shop Scheduling Problem Using A Hybrid Multi-Objective Backtracking Search Algorithm. Journal of Cleaner Production 144 (2017) 228-238.

[11]. L. Yin, X. Li, C. Lu and L. Gao. 2016. Energy-Efficient Scheduling Problem Using an Effective Hybrid Multi-Objective Evolutionary Algorithm, Sustainability, 8, 1268; doi: 10.3390/su8121268.

[12]. K. Deb, 2001. Multi-objective optimization using evolutionary algorithms: Vol. 16. John Wiley & Sons

[13]. G. Mavrotas. 2009. Effective Implementation of the □-constraint Method in Multi-Objective Mathematical Programming Problems. Applied Mathematics and Computation. 213: 455–465.

[14]. Subramanian, A., Battarra, M. & Potts, C. 2014. An iterated local search heuristic for the single machine total weighted tardiness scheduling problem with sequence-dependent setup times. *International Journal of Production Research*. 52(9), 2729-2742.

[15]. Xu, H., Lu, Z. & Cheng, T. 2014. Iterated local search for single-machine scheduling with sequence-dependent setup times to minimize total weighted tardiness. *Journal of Scheduling*, 17(3), 271-287.

[16]. González, M.A., Palacios, J.J., Vela, C.R. & Hernández-Arauzo, A. 2017. Scatter search for minimizing weighted tardiness in a single machine scheduling with setups. *Journal of Heuristics*, 23(2-3), 81-110. https://doi.org/10.1007/s10732-017-9325-1.

[17]. Tasgetiren, M. F., Pan, Q-K, Kizilay, D. & Gao, K. 2016. A variable block insertion heuristic for the blocking flowshop scheduling problem with total flowtime criterion. Algorithms 9(4): 71.

[18]. M. Fatih Tasgetiren, Q-K Pan, Y. Ozturkoglu, A.H.L. Chen, 2016. A memetic algorithm with a variable block insertion heuristic for single machine total weighted tardiness problem with sequence dependent setup times, *IEEE Congress on Evolutionary Computation* CEC 2016: 2911-2918.

[19]. M. Fatih Tasgetiren Quan-Ke Pan, Damla Kizilay, Mario C. Vélez-Gallego, 2017, A variable block insertion heuristic for permutation flowshops with makespan criterion. *IEEE Congress on Evolutionary Computation* CEC 2017: 726-733.

[20]. Osman I. & Potts C. 1989. Simulated annealing for permutation flow-shop scheduling. OMEGA. 17 (6), 551–557

[21]. M. Nawaz, Jr. E. E. Enscore, I. Ham, 1983. A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega*, 11(1), 91-95.

[22]. J. Dubois-Lacoste, F. Pagnozzi, T. Stützle,, 2017. An iterated greedy algorithm with optimization of partial solutions for the makespan permutation flowshop problem. *Computers and Operations Research*. 81: 160–166.

[23]. Q-K. Pan, L. Wang, B. Qian, 2009, A novel differential evolution algorithm for bi-criteria no-wait flow shop scheduling problems, Computers & Operations Research 36 (8), 2498-2511.

[24]. Taillard E, Benchmarks for basic scheduling problems. European Journal of Operational Research 1993;64: 278–85

[25]. C. A. C. Coello, D. A. Van Veldhuizen, and G. B. Lamont. 2002. Evolutionary Algorithms for Solving Multi-Objective Problems. Springer, vol. 242.

[26]. K. C. Tan, C. K. Goh, Y. Yang, T.H. Lee. 2006. Evolving Better Population Distribution and Exploration in Evolutionary Multi-Objective Optimization. European Journal of Operational Research. 171, 463–495.

[27]. E. Zitzler, 1999. Evolutionary Algorithms for Multi Objective Optimization: Methods and Applications: 63. Shaker Ithaca.

A Binary Genetic Algorithm for Solving Bi-Objective Multidimensional Knapsack Problem

Özgür Kabadurmus[1], M. Fatih Tasgetiren[2], Hande Öztop[3], M. Serdar Erdogan[4]

[1,2,4] Department of International Logistics, Yasar University, İzmir, Turkey
{ozgur.kabadurmus,fatih.tasgetiren,mehmet.erdogan}@yasar.edu.tr
[3]Department of Industrial Engineering, Yasar University, İzmir, Turkey
hande.oztop@yasar.edu.tr

Abstract. The multidimensional knapsack problem (MDKP) is an NP-hard problem in combinatorial optimization and it has many practical application areas. However, so far, too little attention has been devoted to multi-objective version of MDKP (MOMDKP) in the literature. In this paper, the bi-objective multidimensional knapsack problem (BOMDKP) is studied and set of non-dominated solutions are found by using a binary genetic algorithm (BGA) with an external archive. The proposed algorithm is tested on various benchmark problems (bi-objective instances with 100, 250, 500 and 750 items) and compared with the other multi-objective algorithms, such as MOEA/D and MOFPA. We observed that true Pareto solutions set provided by Ziztler and Laumans for 500 items and 2 knapsacks contains 30 dominated solutions. For this reason, the true Pareto optimal solutions of the benchmark problems are found for all problem instances using improved augmented epsilon constraint (AUGMECON2) method. The results of the proposed BGA are compared with the true Pareto optimal solutions to show the effectiveness of it. The results show that the proposed BGA algorithm is very competitive when it is compared with the best performing algorithms in the literature.

Keywords: Heuristic optimization, multidimensional knapsack problem, multi-objective optimization, genetic algorithm, augmented epsilon constraint method.

1 Introduction

Knapsack problem (KP) is a well-known combinatorial optimization problem which is to select a subset of items from the set $I = \{1,\ldots,n\}$. Each item i has an associated profit p_i, and weight w_i. The aim of the problem is to maximize the total profit of selected items without exceeding the total weight limit b.

In contrast to the one dimensional KP, Multi-Dimensional Knapsack Problem (MDKP) aims to satisfy more than one constraints. MDKP can be applied on many real life cases such as stock-cutting, project selection, cargo loading [1], shelf space allocation in retail stores [2], scheduling of computer programs [3]. Single-objective MDKP has been investigated in many studies in the previous years. Liu et al. [4] applied a binary differential search algorithm, which uses Brownian motion like random search for solution generation and a feasible solution generation strategy. Meng and Pan [5] proposed an improved fruit fly optimization algorithm employing balance exploitation and exploration for MDKD. Moreover, Meng et al. [6] developed an improved migrating birds optimization algorithm. Chih [7] developed a self-adaptive check and repair operator-based particle swarm optimization (SACRO-PSO) in which pseudo-utility ratio dynamically changes. Chih [8] proposed a three ratio self-adaptive check and repair operator based on particle swarm optimization (3RSACRO-PSO), where substitute pseudo-utility ratios change, systematically. Furthermore, Gorski et al. [9] formulated a greedy algorithm to solve MDKP.

The aim of the multi-objective combinatorial optimization (MOCO) is to find a set of Pareto optimal solutions for multiple conflicting objectives instead of finding a single optimal solution for one objective. Selection of transportation investment alternatives [10], capital budgeting [11], planning the recovery of contaminated light station facilities [12] are some application areas of multi-objective knapsack problem (MOKP). Various studies are conducted on MOKP in KP literature. Visee et al. [13] applied branch and bound approach, while Gandibleux and Freville [14] uses Tabu Search to solve bi-objective KP. Vianna and Arroyo [15] developed a Greedy Randomized Adaptive Search Procedure (GRASP), which finds solutions according to defined preference vector and applies local search to find solutions at each iteration. Lust and Teghem [16] applied two-phase Pareto local search (2PPLS). Kim et al. [17] presented quantum-inspired multi-objective evolutionary algorithm (QMEA). Moreover, Lu and Yu [18] proposed an adaptive population multi-objective quantum-inspired evolutionary algorithm (APMQEA) in which the population size adaptively changes. Gao et al. [19] presented a quantum-inspired artificial immune system (MOQAIS). Cerqueus et al. [20] combined a set of existing branching heuristic to solve bi-objective KP. At each node, the most appropriate heuristics is selected and used with an adaptive selection strategy. Moreover, Mansour and Alaya [21] introduced an indicator-based ant colony optimization (IBACO). Figueira et al. [22] proposed four different exact solution methods for exact solution of MOKP up to seven objectives. Rong and Figueira [23] presented one traditional and a hybrid dynamic programming (DP) algorithms for finding true Pareto optimal solutions of MOKP. Laumans et al. [24] proposed the adaptive ε-constraint method for MOKP. Bazgan et al. [25] proposed a DP algorithm, which uses a dominance relation to eliminate the solutions cannot lead to a new non-dominated criterion vector. Figueira et al. [26] proposed an algorithmic improvement method, which makes use of lower and upper bounds in DP. Furthermore Zitzler and Thiele [27] and Zitzler and Thiele [28] proposed strength Pareto evolutionary algorithm (SPEA), which stores Pareto optimal solutions externally and implements clustering to reduce the number of Pareto optimal solutions to solve MOKP. Zitzler et al. [29] proposed SPEA2 employing an improved fitness assignment technique and new density based selection and archive truncation strategies. In [29], both SPEA and SPEA2 are tested on knapsack problem instances and SPEA2 outperformed SPEA. Furthermore, Zhang and Li [30] developed a multi-objective evolutionary algorithm based on decomposition (MOEA/D) for MOKP.

Multi-objective multidimensional knapsack problem (MOMDKP) is a developing area in the knapsack literature and a very few studies have investigated MOMKDP. To the best of our knowledge, only Zouache et al. [31], Mavrotas et al. [32], Mavrotas et al. [33] and Florios et al. [34] worked on MOMDKP. Zouache et al. [31] introduced a novel multi-objective algorithm, which combines particle swarm optimization (PSO) and firefly algorithm (FA) and, tested the performance of the algorithm on benchmark instances taken from the library published by Zitzler and Thiele [35]. Florios et al. [34] developed an exact multi-criteria branch and bound algorithm for MOMDKP with three objective and three constraints. Both [32] and [33] proposed a heuristics algorithm exploiting "core concept" which was firstly introduced by Balas and Zemel [36]. The core concept considers only a core of items with medium efficiency (p_i/w_i) to reduce the complexity of the problem. While items with high efficiencies are selected, none of the items with low efficiencies is selected. The core concept helps to reduce the search space of the problem and thereby, reduce the complexity. Mavrotas et al. [32] combined branch-and-bound algorithm with an extension of core concept to solve bi-objective MDKP. The algorithm of Mavrotas et al. [33] is not limited to two objective and can handle problems with more than two objectives.

In this study, we propose a simple binary genetic algorithm (BGA) with external archive for solving bi-objective multi-dimensional knapsack problem (MOMDKP). Our proposed methodology is tested over various benchmark problems and the performance of our algorithm is compared to multi-objective firefly algorithm with particle swarm optimization (MOFPA) of Zouache et al. [31] and MOEA/D of Zhang and Li [30]. It should be noted that we also found true Pareto optimal solutions of the benchmark instances by using Improved Augmented Epsilon Constraint (AUGMECON2) proposed by Mavrotas and Florios [37] and demonstrated the efficiency of our algorithm.

The rest of this paper organized as follows: Section 2 describes the problem. Section 3 presents the proposed algorithm. Section 4 discusses the results of the study. Finally, Section 5 concludes the study.

2 Problem formulation

The 0-1 MOMDKP considers selecting a subset of items from a given set of items such that two objectives are maximized without exceeding a set of knapsack capacity constraints. The mathematical model of the problem is defined below. In this formulation, number of items and number of constraints can be different from each other. Although some studies in the literature (such as [27], [28] and [35]) considered the same number of constraints as the number of objective functions, the mathematical model presented here has different numbers for objectives and constraints. Thus, it presents a more general formulation of the problem.

$Notations$:

$n: number\ of\ items$
$m: number\ of\ objectives$
$m: number\ of\ constraints$
$b_i: capacity\ limitation\ of\ constraint\ i$
$c_j^k: profit\ of\ item\ j\ in\ objective\ k$
$a_{ij}: weight\ of\ item\ j\ in\ constraint\ i$

$Mathematical\ formulation$:

$$\begin{aligned}
&ma\,x \quad z_1(x_1, x_2, \dots, x_n) = \textstyle\sum_{j=1}^{n} c_j^1\, x_j \\
&\qquad\qquad\qquad\qquad \vdots \\
&ma\,x \quad z_k(x_1, x_2, \dots, x_n) = \textstyle\sum_{j=1}^{n} c_j^k\, x_j \qquad (1)\\
&\qquad\qquad\qquad\qquad \vdots \\
&ma\,x \quad z_p(x_1, x_2, \dots, x_n) = \textstyle\sum_{j=1}^{n} c_j^p\, x_j \\
&s.t. \\
&\quad \textstyle\sum_{j=1}^{n} a_{ij}\, x_{ij} \ \leq\ b_i\,, \qquad i = 1, \dots, m \qquad (2)\\
&\quad x_j\ \epsilon\,\{0,1\}\,, \qquad\quad j = 1, \dots, n \qquad (3)
\end{aligned}$$

In the mathematical model, the total profit of each objective function is maximized in (1). In this multi-objective model, there can be p different objective functions. Constraint (2) guarantees that total weight of all selected items do not exceed the knapsack capacities. Constraint (3) ensures that all decision variables are binary.

As our problem is a multi-objective problem, we present the dominance relation concepts [40] below that will be used when solving the MOMDKP:

- **Definition 1: Dominance relation**. A solution x_i dominates another solution x_j if the two following conditions are satisfied (denoted as $x_i \succ x_j$):
  - $\forall p \in 1, .., P; f_p(x_i) \geq f_p(x_j)$
  - $\exists p \in 1, .., P; f_p(x_i) > f_p(x_j)$

  A solution x_i weakly dominates another solution x_j (denoted as $x_i \succcurlyeq x_j$) if :
  - $\forall p \in 1, .., P; f_p(x_i) \geq f_p(x_j)$

  A solution x_i is indifferent to another solution x_j (denoted as $x_i \backsim x_j$) if :
  - $\forall p \in 1, .., P; f_p(x_i) \ngeq f_p(x_j)\ \wedge\ f_p(x_j) \ngeq f_p(x_i)$
- **Definition 2: Non-dominated set**: Amongst a set of solutions X, the non-dominated set of solutions are the elements of the set X^* non-dominated by any element of the set X.

- **Definition 3: Pareto-optimal set**: The non-dominated set of the entire feasible search space S is called the Pareto-optimal solutions.

There are common solution methods for multi-objective problems such as sequential optimization, goal programming, weighting method and ε-constraint method. In this study, we prefer to use improved augmented ε-constraint method (AUGMECON2), as it generates only Pareto-optimal solutions [37]. In Pareto-optimal solutions, any objective function cannot be improved without worsening another objective function. In improved augmented ε-constraint method, one of the objective functions is optimized, while other objective functions are defined by constraints. Dissimilar to the standard ε-constraint method, slack/surplus variables are included in these objective function constraints by converting them to equalities. These variables are also defined as second term in the objective function to ensure the Pareto-optimality. More details on AUGMECON2 can be found in [37].

3 Bi-objective BGA

Genetic algorithms (GA) are search heuristics based on the biological process of natural selection and evolution [38]. In GAs, individuals with decision variables in D dimensions are encoded into chromosomes to obtain an initial population that should be evolved over generations. At each generation, two individuals are chosen and mated from the population. Then, two individuals are crossed over to generate new solutions called offspring or child. Some individuals are mutated to escape from local minima. Ultimately, offspring population is added to parent population in order to select new individuals for the next generation.

In Fig. 1, outline of BGA is given for the MOMDKP, and this is detailed in the following parts of this section. First, the initial population of size NP is constructed randomly. At each generation, for each individual in the population, another individual from the population is randomly selected to produce an offspring through uniform crossover. Once a new solution is obtained by crossover operator, this solution is evaluated and the superiority of feasibility (SF) rule by [39] is used to update both parent population and the archive set Ω. This is repeated for each individual in the population. Then, a local search proposed in [40] is applied to each individual in the population. Again, new offspring is evaluated and the SF rule is used to update both parent population and the archive set Ω. Furthermore, each individual in the population is mutated by changing binary string to one or zero randomly with a small probability. This procedure is repeated until the stopping criterion is achieved.

```
Initialize parameters
Initialize population
Evaluate population
While (not termination) do
    Crossover
    Local Search
    Selection
    Mutation
Endwhile
Report nondominated solution set
```

Fig. 1. Binary genetic algorithm

Each individual is a solution of the MKP, which is represented by an n-bit binary string. In the BGA algorithm, solutions are represented by a string of zero or one. If item j is selected into the knapsack, then, $j = 1$; otherwise, $j = 0$.

An initial population of NP individuals are randomly established with an equal probability of 0.5. In other words, if a uniform random number $r(0,1)$ is less than 0.5, the decision variable is assigned to 1. Otherwise, it is assigned to 0. In addition, crossover ($pC[i]$) and mutation ($pM[i]$) probabilities are assigned to 0.5 and 0.01 initially. The outline of the initial population is given in Fig. 2.

```
$for\ i = 1\ to\ NP\ do$
    $pC[i] = 0.5$
    $pM[i] = 0.01$
    $for\ j = 1\ to\ n\ do$
        $if\ (r < 0.5)\ then\ do$
            $x_{ij} = 1$
        $else$
            $x_{ij} = 0$
    $endif$
$endfor$
```

Fig. 2. Initial population

Offspring population is obtained through a simple uniform crossover operator. In crossover operator, we generate an offspring for each individual x_i in the population in such a way that another individual is selected from the population randomly, say x_k and a uniform crossover is applied with a crossover probability $pC[i]$ as follows:

$$x_{ij}^{*} = \begin{cases} x_{ij} & if\ r_{ij} \leq pC[i] \\ x_{kj} & otherwise \end{cases} \qquad \forall j \in 1,..,n;\ \forall i \in 1,..,NP \tag{4}$$

where r_{ij} is a uniform random number in $U(0,1)$ and $CR[i]$ is the crossover probability, which is drawn from the unit normal distribution $N(0.5,0.1)$ for each individual x_i in the population. Note that crossover probability $CR[i]$ will be mostly around 0.5 and deviating from 0.2 to 0.8 occasionally. After crossover operator, parent population and the archive set is updated according to $SF\_Domination$ rule explained in Fig. 5. The outline of the procedure is given in Fig. 3.

```
$for\ i = 1\ to\ NP\ do$
    $x_i = \begin{cases} x_i^* & if\ (x_i^*\ is\ SF\_Dominate\ to\ x_i) \\ x_i & otherwise \end{cases}$
$endfor$
```

Fig. 3. Update parent population and archive set

After having carried out crossover for all individuals in the population, we mutate the population with a small mutation probability as follows:

$$x_{ij} = \begin{cases} x_{ij} = 0 & if\ r_{ij} \leq pM[i] \\ x_{ij} = 1 & otherwise \end{cases} \tag{5}$$

where r_{ij} is a uniform random number in $U(0,1)$ and $pM[i]$ is the mutation probability, which is drawn from the unit normal distribution $N(0.05,0.01)$ for each individual x_i in the population.

The following binary local search is presented in [40], which is shown that it is very effective for single objective MDKP by using a design of experiment. We apply a binary local search to the parent population in this step. For each individual in the population, we choose three items randomly. Then, we flip them to 0 if the item is 1 or to 1 if the item is 0. The local search is given in Fig. 4.

Note that we do not repair solutions if they are infeasible because search can benefit from infeasible solutions [39]. Since the problem on hand is a constrained optimization problem, we employ the superiority of feasible solution (SF) rule [39]. Regarding the constrained multi-objective optimization, when comparing two individuals, the situation is somewhat different [39]. Three cases can be observed: 1) one is feasible, the other is not; 2) both are infeasible: and 3) both solutions are feasible. For the constrained multi-objective optimization on hand, we modify SF rule in NSGA-II of Deb et al. [41] to make the definition of *domination* between two solutions as follows: An individual x^* is considered to be $SF\_Domination$ to an individual x under the following conditions:

1) Individual x^* is feasible and individual x is infeasible. Hence, x^* is preferred to x by $SF\_Domination$ rule.
2) Both individuals are infeasible, but individual x^* has a smaller constraint violation. Hence, x^* is preferred to x as $SF\_Domination$
3) Both individuals, x^* and x are feasible and individual x^* dominates individual x. Since x^* dominates x ($x^* \succ$ x), update the external archive set Ω by $SF\_Domination$ rule.

```
for $i = 1$ to $NP$ do
    $x^* = x$
    do
        $a = rand()\%n$
        $b = rand()\%n$
        $c = rand()\%n$
    while $(a = b = c)$
    flip $x^*_{i,a}, x^*_{i,b}$ and $x^*_{i,c}$ to either 0 or 1
    compute objective functions of $x_i^*$
      $x_i = \begin{cases} x_i^* & if\ (x_i^*\ is\ SF\_Dominate\ to\ x_i) \\ x_i & otherwise \end{cases}$
endfor
```

Fig. 4. Binary local search

By using this idea above, we develop a selection procedure, denoted by $SF\_Domination$ rule, which is given in Fig. 5.

```
for $i = 1$ to $NP$ do
    if $(x_i^*.violation = 0$ and $x_i.violation > 0)$
        $x_i = x_i^*$
    else
        if $(x_i^*.violation > 0$ and $x_i.violation > 0)$
            if $(x_i^*.violation < x_i.violation)$
                $x_i = x_i^*$
            else
                if $(x_i^*.violation = 0$ and $x_i.violation = 0)$
                    if $(x_i^* \succ x_i)$
                        $x_i = x_i^*$; update the archive set $\Omega$
                    else
                        $x_i = x_i$
                    endif
                endif
            endif
        endif
    endif
endfor
```

Fig. 5. $SF\_Domination$ rule

An archive set Ω is used to store the non-dominated solutions during the optimization process. This archive set should be updated with non-dominated solutions in order to approximate the Pareto-optimal solutions. When a new non-dominated solution is obtained, it should be added to the archive set Ω and any member dominated by the new non-dominated solution should be removed. In order to update the archive set Ω, Pan et.al. [42] proposed an effective method for updating the archive set as follows: The non-dominated solutions in Ω are stored in decreasing order of their first objective function values. Then, their second objective values will be in increasing order. The procedure for updating the archive set Ω can be summarized as follows:

Step 1. Archive size is $m = |\Omega|$ and $\Omega = \{a_1, a_2, .., a_m\}$. Initially, Ω is empty and the first non-dominated solution x will be added to the first position in Ω. Let $k = 1$.

Step 2. Find a most suitable position pos for the next individual x in the archive set Ω by the following procedure:

```
$do\{$
  $j = \lfloor (k+m)/2 \rfloor$
  $if\ \left(f_1(x) = f_1(a_j)\right)\ then\ j = j$
  $elseif\ \left(f_1(x) > f_1(a_j)\right)\ then\ m = j - 1$
  $else\ \ k = j + 1$
$while(k \le m)$
```

Step 3. When comparing $f_1(x)$ with $f_1(a_j)$, following cases may occur:

```
$Case\ 1.\ if\ \left(f_1(x) = f_1(a_j)\right)\ and\ if\ \left(f_2(x) > f_2(a_j)\right) then\ pos = j$
$Case\ 2.\ if\ \left(f_1(x) > f_1(a_j)\right)$
    $if\ \ j = 1\ \ then\ pos = j\ \ and\ m = m + 1$
    $if\ \ j > 1\ and\ \ if\ \left(f_2(x) > f_2(a_{j-1})\right) then\ pos = j\ and\ m = m + 1$
$Case\ 3.\ if\ \left(f_1(x) < f_1(a_j)\right)\ and\ if\ \left(f_2(x) > f_2(a_j)\right) then\ pos = j + 1\ and\ m = m + 1$
```

If any of cases above is satisfied, solution x is added to position pos, but all solutions dominated by x in Ω should be removed. The following procedure removes the dominated solutions from Ω:

```
Step 1.   $If\ (pos = m)\ then\ go\ to\ Step\ 4$
Step 2.   $Let\ pos = pos + 1.\ If\ f_2(a_{pos}) \le f_2(x)\ then\ remove\ a_{pos};\ otherwise\ go\ to\ Step\ 4$
Step 3.   $if (pos < m)\ then\ go\ to\ Step\ 2$
Step 4.   $\Omega = report\ non - dominated\ solutions$
```

5 Computational Results

To evaluate the performance of our algorithm, we used the benchmark problem instances of Zitzler and Thiele [35]. We tested our algorithm on four different benchmark instances with 100, 250, 500, and 750 items having 2 knapsacks. We executed 30 replications with different random seeds for each instance. Non-dominated solution sets were found in all replications and then these sets were united into a single Pareto optimal set according to the dominance relation. We applied this procedure to the results of 30 replications of MOFPA of Zouache et al. [31] and MOEA/D of Zhang and Li [30] and compared them with our BGA algorithm. We also compared our algorithm with the true Pareto optimal solutions that we obtained by using AUGMECON2. Note that, the true Pareto optimal solutions are not given in Zitzler and Thiele [35]. We found the true Pareto optimal solutions using AUGMECON2.

Our BGA algorithm was coded in Visual C++ 13 and carried out on an Intel$^{(R)}$ Core$^{(TM)}$ i7-2600 CPU with 3.40GHz PC with 8.00GB memory. The population size is taken as *NP*=100. It was carried out 30 replications for each benchmark problem. MOFPA and MOEA/D results are taken from [31]. Each replication is terminated after reaching the the maximum CPU time, i.e., $Tmax = 100n$ millisecond.

As we find the true Pareto-optimal sets for the benchmark instances, we use the following performance measures to compare the performances of the three algorithms (BGA, MOFPA and

MOEA/D). Note that N refers to the non-dominated solution set of any algorithm and P refers to the true Pareto-optimal set.

- Ratio of the Pareto-optimal solutions found: $R_p = |N \cap P|/|P|$
- Inverted Generational Distance [43]: $IGD = \sum_{v \in P} d(v, N)/|P|$, where $d(v, N)$ denotes the minimum Euclidean distance between v and the solution in N. The low IGD value means that set N is very close to set P.
- Distribution Spacing [44]: $DS_N = \frac{\left[\frac{1}{|N|}\sum_{i \in N}(d_i - \bar{d})^2\right]^{1/2}}{\bar{d}}$, where $\bar{d} = \frac{\sum_{i \in N} d_i}{|N|}$, d_i is the minimum Euclidean distance between solution i and its closest neighbour in N. Low spacing value indicates that the solutions in N are evenly distributed.

Tables 1, 2, and 3 report the results of R_p, IGD and DS measures of BGA, MOFPA and MOEA/D algorithms, respectively. Since [31] did not report the results of MOFPA and MOEA/D for 100 items scenario, we compared our BGA algorithm with the true Pareto optimal set and we calculate the average values in all tables using the instances with 250, 500 and 750 items. As shown in Table 1, our algorithm is able to find 97% of the solutions in the true Pareto set for the scenario with 100 items. According to R_p metric, our algorithm outperforms MOEA/D and MOFPA. As given in Table 3, MOEA/D could not find any true Pareto optimal solution in all scenarios. For the benchmark instance with 250 items, our algorithm finds 67% of the solutions in the true Pareto optimal set, whereas MOFPA finds 12% of the true Pareto optimal solutions. For the benchmark instance with 500 items, BGA finds 32% of the solutions in the true Pareto set, while MOFPA finds 1% of the true Pareto optimal solutions. For the benchmark instance with 750 items, our algorithm finds 3% of the solutions in the true Pareto optimal set, whereas MOFPA finds 1% of the true Pareto optimal solutions.

Table 1. Comparison of BGA and true Pareto

Instance	R_p	IGD	DS$_{\text{BGA}}$
100 Items	0.97	0.43	0.94
250 Items	0.67	2.98	1.44
500 Items	0.32	38.15	0.99
750 Items	0.03	184.98	1.4
Average	**0.34**	**75.37**	**1.28**

Table 2. Comparison of MOFPA and true Pareto

Instance	R_p	IGD	DS$_{\text{MOFPA}}$
250 Items	0.12	7.63	0.85
500 Items	0.01	15.95	1.04
750 Items	0.01	105.77	1.29
Average	**0.05**	**43.12**	**1.06**

Table 3. Comparison of MOEAD and true Pareto

Instance	R_p	IGD	DS$_{\text{MOEAD}}$
250 Items	0.00	27.55	0.88
500 Items	0.00	129.76	0.68
750 Items	0.00	336.71	0.73
Average	**0.00**	**164.67**	**0.76**

In terms of IGD metric, MOFPA has the lowest IGD value on average, whereas MOEAD has the highest IGD values for all scenarios. However, BGA also has low IGD values, especially for the instances with 100 and 250 items. Note that BGA has the lowest IGD value for the instance with 250 items and it has very low (0.43) IGD value for the instance with 100 items, which indicate that the BGA provides very close approximations to the Pareto optimal solution sets. In terms of distribution spacing, the solutions of MOEAD are more evenly distributed than the other algorithms due to the lowest average DS value.

However, the DS values of all algorithms are very close to each other. Therefore, it can be said that BGA also has a good distribution as the difference between average DS values of BGA and MOEAD is very low.

6 Conclusion

Bi-objective Multidimensional Knapsack Problem is an extension of the well-known single objective knapsack problem. It is an NP-hard combinatorial optimization problem. The problem aims to find a Pareto optimal set of two maximization objective functions by selecting a subset of items without exceeding the knapsack capacities. Despite the rich literature of the multi-objective single dimensional and single objective multidimensional knapsack problems, very few studies have investigated multi-objective multidimensional knapsack problem.

In this study, we proposed a new solution method, Binary GA with External Archive to solve Bi-objective Multidimensional Knapsack Problem. The proposed algorithm has been tested on four different benchmark instances that was published by [35]. We also compared our results with the ones of MOFPA of [31] and MOEA/D of [30]. Our algorithm outperforms MOFPA and MOEA/D in terms of the number of Pareto optimal solutions found. Also, the true Pareto optimal solutions of the benchmark problems were found for all problem instances using Improved Augmented Epsilon Constraint (AUGMECON2) method. While our algorithm finds almost all true Pareto solutions for small instances, it finds close solutions to true Pareto in medium and large sized instances.

For future research, the proposed algorithm of this paper to solve Bi-objective Multidimensional Knapsack Problem can be improved in many ways. The core concept can be added or local search algorithms can be included to improve the performance of our Binary GA algorithm. In addition, the algorithm can be tested on three objective multidimensional knapsack problems. All data files, computational results and Pareto charts are provided as supplementary materials, and can be found in https://okabadurmus.yasar.edu.tr/research/meta2018-solutions/

References

[1] B. Haddar, M. Khemakhem, S. Hanafi and C. Wilbaut, A hybrid quantum particle swarm optimization for the Multidimensional knapsack problem, *Engineering Applications of Artificial Intelligence,* 55 (2016) 1–13.

[2] M. H. Yang, An efficient algorithm to allocate shelf space. European Journal of Operational Research, 131(1) (2001) 107-118.

[3] H. Kellerer, U. Pferschy and D. Pisinger, Knapsack Problems, Springer-Verlag, Berlin, 2004.

[4] J. Liu, C. Wu, J. Cao, X. Wang and K. L. Teo, A Binary differential search algorithm for the 0–1 multidimensional knapsack problem. Applied Mathematical Modelling, 40 (23) (2016) 9788-9805.

[5] T. Meng and Q. K. Pan, An improved fruit fly optimization algorithm for solving the multidimensional knapsack problem. Applied Soft Computing, 50 (2017) 79-93.

[6] T. Meng, J. H. Duan and Q. K. Pan, An improved migrating birds optimization for solving the multidimensional knapsack problem. in IEEE 29th Control And Decision Conference, (2017) 4698-4703.

[7] M. Chih, Self-adaptive check and repair operator-based particle swarm optimization for the multidimensional knapsack problem. Applied Soft Computing, 26 (2015) 378-389.

[8] M. Chih, Three pseudo-utility ratio-inspired particle swarm optimization with local search for multidimensional knapsack problem. Swarm and Evolutionary Computation, (2017).

[9] J. Gorski, L. Paquete, and F. Pedrosa, Greedy algorithms for a class of knapsack problems with binary weights. Computers & Operations Research, 39(3) (2012) 498-511

[10] J. Y. Teng, and G. H. Tzeng, A multiobjective programming approach for selecting non-independent transportation investment alternatives. Transportation Research Part B: Methodological, 30 (4) (1996) 291-307.

[11] M. J. Rosenblatt and Z. Sinuany-Stern, Generating the discrete efficient frontier to the capital budgeting problem. Operations Research, 37 (3) (1989) 384-394.

[12] L. Jenkins, A bicriteria knapsack program for planning remediation of contaminated lightstation sites. European Journal of Operational Research, 140 (2) (2002) 427-433.

[13] M. Visée, J. Teghem, M. Pirlot and E. L. Ulungu, Two-phases method and branch and bound procedures to solve the bi–objective knapsack problem. Journal of Global Optimization, 12 (2) (1998) 139-155.

[14] X. Gandibleux and A. Freville, Tabu search based procedure for solving the 0-1 multiobjective knapsack problem: the two objectives case. Journal of Heuristics, 6 (3) (2000) 361-383.

[15] D. S. Vianna and J. E. C. Arroyo, A GRASP algorithm for the multi-objective knapsack problem. in IEEE 24th International Conference of the Chilean Computer Science Society (2004) 69-75.

[16] T. Lust and J. Teghem, The multiobjective multidimensional knapsack problem: a survey and a new approach. International Transactions in Operational Research, 19(4) (2012) 495-520.

[17] Y. Kim, J. H. Kim and K. H. Han, Quantum-inspired multiobjective evolutionary algorithm for multiobjective 0/1 knapsack problems. In IEEE Congress on Evolutionary Computation, (2006) 2601-2606.

[18] T. C. Lu and G. R. Yu, An adaptive population multi-objective quantum-inspired evolutionary algorithm for multi-objective 0/1 knapsack problems. Information Sciences, 243 (2013) 39-56.

[19] J. Gao, G. He, R. Liang and Z. Feng, A quantum-inspired artificial immune system for the multiobjective 0–1 knapsack problem. Applied Mathematics and Computation, 230 (2014) 120-137.

[20] A. Cerqueus, X. Gandibleux, A. Przybylski and F. Saubion, On branching heuristics for the bi-objective 0/1 unidimensional knapsack problem. Journal of Heuristics, 23(5) (2017) 285-319.

[21] I. B. Mansour and I. Alaya, Indicator based ant colony optimization for multi-objective knapsack problem. Procedia Computer Science, 60 (2015) 448-457.

[22] J. R. Figueira, G. Tavares and M. M. Wiecek, Labeling algorithms for multiple objective integer knapsack problems. Computers & Operations Research, 37(4) (2010) 700-711.

[23] A. Rong and J. R. Figueira, Dynamic programming algorithms for the bi-objective integer knapsack problem. European Journal of Operational Research, 236(1) (2014) 85-99.

[24] M. Laumanns, L. Thiele and E. Zitzler, An efficient, adaptive parameter variation scheme for metaheuristics based on the epsilon-constraint method. European Journal of Operational Research, 169(3) (2006) 932-942.

[25] C. Bazgan, H. Hugot and D. Vanderpooten, Solving efficiently the 0–1 multi-objective knapsack problem. Computers & Operations Research, 36(1) (2009) 260-279.

[26] J. R. Figueira, L. Paquete, M. Simões, and D. Vanderpooten, Algorithmic improvements on dynamic programming for the bi-objective {0, 1} knapsack problem. Computational Optimization and Applications, 56(1) (2013) 97-111.

[27] E. Zitzler, and L. Thiele, An evolutionary algorithm for multiobjective optimization: The strength pareto approach, (1998).

[28] E. Zitzler and L. Thiele, Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. IEEE transactions on Evolutionary Computation, 3(4) (1999) 257-271.

[29] E. Zitzler, M. Laumanns, and L. Thiele, SPEA2: Improving the strength Pareto evolutionary algorithm. TIK-report, 103 (2001).

[30] Q. Zhang and H. Li, MOEA/D: A multiobjective evolutionary algorithm based on decomposition. IEEE Transactions on evolutionary computation, 11(6) (2007) 712-731.

[31] D. Zouache, A. Moussaoui, and F. B. Abdelaziz, A cooperative swarm Intelligence algorithm for multi-objective discrete optimization with application to the knapsack problem. European Journal of Operational Research, 264(1) (2018) 74-88.

[32] G. Mavrotas, J. R. Figueira and K. Florios, Solving the bi-objective multi-dimensional knapsack problem exploiting the concept of core. Applied Mathematics and Computation, 215 (7) (2009) 2502-2514.

[33] G. Mavrotas, K. Florios and J. R. Figueira, An improved version of a core based algorithm for the multi-objective multi-dimensional knapsack problem: a computational study and comparison with meta-heuristics. Applied Mathematics and Computation, 270 (2015) 25-43.

[34] K. Florios, G. Mavrotas, and D. Diakoulaki, Solving multiobjective, multiconstraint knapsack problems using mathematical programming and evolutionary algorithms. European Journal of Operational Research, 203(1) (2010) 14-21.

[35] E. Zitzler and L. Thiele, Multiobjective optimization using evolutionary algorithms—A comparative case study. In Proceedings of international conference on parallel problem solving from nature, (1998) 292–301.

[36] E. Balas, and E. Zemel, An algorithm for large zero-one knapsack problems. Operations Research, 28 (5) (1980) 1130-1154.

[37] G. Mavrotas and K. Florios, An improved version of the augmented ε-constraint method (AUGMECON2) for finding the exact pareto set in multi-objective integer programming problems. Applied Mathematics and Computation, 219(18) (2013) 9652-9669.

[38] D. E. Goldberg, Genetic algorithms in search, optimization, and machine learning. Reading, MA: Addison-Wesley; (1989).

[39] K. Deb, An efficient constraint handling method for genetic algorithms. Computer methods in applied mechanics and engineering, 186(2-4) (2000) 311-338.

[40] L. Wang, X-L Zheng, S-Y Wang, A novel binary fruit fly optimization algorithm for solving the multidimentional knapsack problem, Knowledge-Based Systems 48 (2013) 17-23

[41] K. Deb, A. Pratap, S. Agarwal, and T. A. M. T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE transactions on evolutionary computation, 6(2) (2002) 182-197.

[42] Q-K. Pan, L. Wang, B. Qian, 2009, A novel differential evolution algorithm for bi-criteria no-wait flow shop scheduling problems, Computers & Operations Research 36 (8), 2498-2511.

[43] C. A. C. Coello, D. A. Van Veldhuizen, and G. B. Lamont. 2002. Evolutionary Algorithms for Solving Multi-Objective Problems. Springer, vol. 242.

[44] K. C. Tan, C. K. Goh, Y. Yang, T.H. Lee. 2006. Evolving Better Population Distribution and Exploration in Evolutionary Multi-Objective Optimization. European Journal of Operational Research. 171, 463–495.

An Agent-Based Label Propagation Algorithm for Community Detection

A.M. Fiscarelli[1,2], M.R. Brust[2], G. Danoy[3] and P. Bouvry[2,3]

[1] C2DH, University of Luxembourg, 11 Porte des Sciences, Esch-sur-Alzette
antonio.fiscarelli@uni.lu
[2] SnT, University of Luxembourg, 29 Avenue John F. Kennedy, 1855 Luxembourg
antonio.fiscarelli@uni.lu, matthias.brust@uni.lu
[3] FSTC-CSC-ILIAS, University of Luxembourg, 2 Avenue de l'Université L-4365 Esch-sur-Alzette
gregoire.danoy@uni.lu, pascal.bouvry@uni.lu

1 Introduction

Many complex systems such as social networks, the World Wide Web, bio-molecular interactions and multi-agent systems, can be represented as networks. Often, they exhibit a community structure: nodes inside a community are densely connected to each other and loosely connected to any other node of the network. Communities play an important role in networks and the evolution of communities can provide relevant information about the dynamics of the system. The objective of a community detection algorithms is to partition similar nodes of the network in a variable number of groups having different sizes.

Several methods have been proposed that use different approaches: modularity optimization, spectral methods or methods based on random walks. In particular, Raghavan [1] proposed a method called label propagation algorithm (LPA) where nodes are assigned an unique label and, at each iteration, each node adopts the label chosen by the majority of its neighbors. After few iterations, the network converges to a steady state and nodes having the same label are grouped into communities. This method is outperformed by some of the other algorithms and can get stuck in local optima, but several variations have been proposed to overcome these problems. Furthermore, this method runs in near linear time and does not require global information of the network, thus it is especially suitable for very large networks or for systems where only local connectivity is available.

The next section will first describe some related works on community detection and, in particular, label propagation algorithms and their variations. Then, our new agent-based label propagation algorithm will be introduced. Finally, we will conclude and propose some ideas for future work.

2 Related work

In this section we give an overview of the existing community detection methods. Girvan and Newman [2] first proposed a divisive hierarchical algorithm that makes use of edge betweenness: it measures the number of shortest paths between all pairs of nodes in the network that include a certain edge. Edges connecting nodes in different communities have high edge betweennes and removing them would enhance the separation of communities. A faster version [3] was proposed: it is a hierarchical agglomerative algorithm that assigns a different community to each node and iteratively merges them to optimize modularity. Blondel, Guillaume, and Lambiotte [4] proposed another hierarchical agglomerative method called Louvain that maximizes modularity. It assigns to each node the community of the neighbor that would bring the best improvement in modularity, merges nodes in the same communities and builds a network where nodes represent the communities just found. The process iterates until improvement no longer occur. Walktrap [5] is a random walk based algorithm that uses the transition probability of random walkers to define a similarity between nodes. In fact, random walkers are more likely to stay within the same community. Infomap [6], in a similar fashion, models flow patterns in networks using the transition probability of random walkers. Newman [7] proposed a spectral method based on the eigenspectrum of the modularity matrix that maximizes modularity. Finally, Reichardt and Stefan Bornhol [8] interpret community detection as minimizing the energy function of a spin model, where communities are defined as spin configurations.

The algorithms just described have several downfalls that make them inefficient on large scale networks or multi-agent systems where only local interaction occurs: they cannot find a solution in a reasonable time, they require global information of the network or they suffer from the resolution limit. Raghavan [1] proposed a method called label propagation algorithm (LPA) which runs in near linear time and uses only the network structure to drive the search. Nodes are initially assigned an unique label and, at each iteration, each node adopts the label chosen by the majority of its neighbors. After few iterations, the network converges to a steady state and nodes having the same label are grouped into communities. This can also be seen as an epidemics model where each node is an infectious agent injecting the network with a different disease (label), or an evolutionary game where each node is an agent that plays a fictitious game by keeping beliefs about neighbors' choices and take an action (choose a label) according to them. Unfortunately, this comes with some drawbacks. LPA gets easily stuck in local optima and is often outperformed by more sophisticated algorithms. Also, for networks where communities are weakly defined, a certain label may ”flood” the network resulting in an a single giant community. Several variations have been proposed to overcome these problems. Clark [9] developed a variation of this method that incorporates modularity optimization to obtain better results, while Liu and Murata [10] integrated a greedy agglomerative algorithm that allows it to escape from local optima. Leung [11] uses a more sofisticated decision rule based on node preference (in this case, node degree) to improve performance and hop attenuation to prevent a label to spread too quickly, while keeping the algorithm scalable. Xie and Szymanski [12] implement a different node preference that is based on the number of common neighbors between nodes and is related to the clustering coefficient. Šubelj and Bajec [13] elaborate two particular strategies called defensive preservation and offensive expansion. These strategies rethink the heuristics proposed by Leung to focus on core nodes and border nodes of communities and are hierarchically combined in a sinergic algorithm. They also show that the choice of node preference and hop attenuation depend on the structure of the network. Xie and Szymanski [14] also proposed a variation of LPA that takes inspiration from the MCL algorithm [15]: each node keeps a list of label distributions that are propagated through the network. An inflation and a cutoff operator are applied to these lists in a decentralized fashion to keep their size small and speedup the algorithm.

3 MemLPA

The method that we would like to propose is called MemLPA, a variation of the original label propagation algorithm where each node is seen as an agent that interacts and shares information with its neighbors and implements memory. In fact, in the basic LPA, nodes choose a label according to the state of its neighborhood at the current iteration, ignoring past states, while in our method agents keep a list of the labels chosen by their neighbors during previous iterations. We also adapt some of the improvements proposed in literature to our method such as neighborhood preference, hop attenuation and conditional update. In the experimental analysis we have found how the implementation of memory can solve some of the issues presented: it can successfully improve performance and can prevent aggressive labels to spread through the whole network. The algorithm can run in near linear time on directed or undirected, weighted or unweighted networks, is decentralized and scalable.

We compare our method against other existing LP variations and show the advantages of implementing memory. We also compare it to other state-of-the-art community detection algorithms. For the analysis, we run all algorithms on the Lancichinetti-Fortunato-Radicchi (LFR) benchmark [16], a very well-known benchmark in literature for community detection, and use the Normalized Mutual Information (NMI) as performance metric [17] and [18]. We also apply the algorithms on a set of real-world networks of different nature and use the modularity measure to evaluate the quality of the solutions.

4 Conclusions and future work

In this paper, we introduce the state-of-the-art of community detection techniques and illustrate their main drawbacks. We discuss the label propagation algorithm with its variations, and show how this method can be translated to a disease spreading model or an evolutionary game and we proposed a new label propagation algorithm that sees nodes as agents interacting with its neighbors and implementing memory. In particular, we aimed at developing a method that follows the swarm intelligence principles:

- Self-organization: by defining the behavior of single robots, a collective behavior emerges
- Decentralized: agents use only local interaction to exchange information and choose actions. Also, they have no global information about the network
- Scalable: the collective behavior does not change when network size increase. Also, it is suitable for parallelization
- Fast: has near-linear time complexity
- Stable: does not oscillate and does not get stuck in local optima
- Adaptive: the algorithm can converge to a new equilibrium if agents are deleted/added

We conducted experiments on artificial networks and real world networks, compared out method against other state-of-the-art community detection algorithms and investigated how the implementation of memory can solve some of the issues described. As possible upcoming work, we intend to investigate the possibility to adapt other variations proposed in literature to MemLPA, perform tests on networks having different structures and find out whether there is a correlation between the structure of the network and any of these variations. We also want to test this method on swarms of UAVs for applications such as area coverage optimization and intrusion of malicious UAVs, where the system can be represented as a dynamic network.

References

1. Raghavan, U.N., Albert, R., Kumara, S.: Near linear time algorithm to detect community structures in large-scale networks. Physical review E **76** (2007) 036106
2. Newman, M.E., Girvan, M.: Finding and evaluating community structure in networks. Physical review E **69** (2004) 026113
3. Clauset, A., Newman, M.E., Moore, C.: Finding community structure in very large networks. Physical review E **70** (2004) 066111
4. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. Journal of statistical mechanics: theory and experiment **2008** (2008) P10008
5. Pons, P., Latapy, M.: Computing communities in large networks using random walks. In: ISCIS. Volume 3733. (2005) 284–293
6. Rosvall, M., Bergstrom, C.T.: Maps of random walks on complex networks reveal community structure. Proceedings of the National Academy of Sciences **105** (2008) 1118–1123
7. Newman, M.E.: Finding community structure in networks using the eigenvectors of matrices. Physical review E **74** (2006) 036104
8. Reichardt, J., Bornholdt, S.: Statistical mechanics of community detection. Physical Review E **74** (2006) 016110
9. Barber, M.J., Clark, J.W.: Detecting network communities by propagating labels under constraints. Physical Review E **80** (2009) 026129
10. Liu, X., Murata, T.: Advanced modularity-specialized label propagation algorithm for detecting communities in networks. Physica A: Statistical Mechanics and its Applications **389** (2010) 1493–1500
11. Leung, I.X., Hui, P., Lio, P., Crowcroft, J.: Towards real-time community detection in large networks. Physical Review E **79** (2009) 066107
12. Xie, J., Szymanski, B.K.: Community detection using a neighborhood strength driven label propagation algorithm. arXiv preprint arXiv:1105.3264 (2011)
13. Šubelj, L., Bajec, M.: Unfolding communities in large complex networks: Combining defensive and offensive label propagation for core extraction. Physical Review E **83** (2011) 036103
14. Xie, J., Szymanski, B.K.: Labelrank: A stabilized label propagation algorithm for community detection in networks. In: Network Science Workshop (NSW), 2013 IEEE 2nd, IEEE (2013) 138–143
15. Dongen, S.: A cluster algorithm for graphs. (2000)
16. Lancichinetti, A., Fortunato, S., Radicchi, F.: Benchmark graphs for testing community detection algorithms. Physical review E **78** (2008) 046110
17. Danon, L., Diaz-Guilera, A., Duch, J., Arenas, A.: Comparing community structure identification. Journal of Statistical Mechanics: Theory and Experiment **2005** (2005) P09008
18. Yang, Z., Algesheimer, R., Tessone, C.J.: A comparative analysis of community detection algorithms on artificial networks. Scientific reports **6** (2016) 30750

Forecasting patients flow at an emergency department

R. Sadeghi[1], J. Oesterle [1], T. Arbaoui [2], F. Yalaoui [2], H. Chehade [1]

1. Opta LP, 2 Rue Gustave Eiffel, 10430 Rosières-prés-Troyes, France
sadeghi@opta-lp.com, oesterle@opta-lp.com, chehade@opta-lp.com

2. Institut Charles Delaunay, LOSI, Université de Technologie de Troyes, 12 Rue Marie Curie, CS 42060, cedex, 10004, Troyes, France
taha.arbaoui@utt.fr, farouk.yalaoui@utt.fr

Keywords: forecasting, emergency department, patient flow, time series, ARIMA models, artificial neural networks

1 Introduction

Forecasting of future events is a sensible and essential activity that has a direct impact on the ability of any organization to e.g. schedule the level of demand and material [3] and plan the necessary resources. In this study we focus on forecasting the number of patients in emergency departments (ED) with the perspective of providing a better management of human resources (doctors, nurses, etc.) to decrease the average waiting time of patients. The complexity of forecasting the patients flow is achieved using the creation of models describing the behaviors of the whole system. A Time series is a sequence of observations collected over time. Time series analysis can provide accurate predictions of ED's volume, length of stay, and acuity [9] and has been proven useful in estimating a future event [5]. Many methods have been developed to model and forecast time series such as Simple Moving Average (SMA), Exponential Smoothing (ES), Autoregressive Integration Moving Average (ARIMA), and Artificial Neural Networks (ANN).

ARIMA is one of the most important and widely used method for building time series forecasting models [4]. The main advantages of ARIMA method is its flexibility in forecasting a large number of time series [6]. But the major drawback of this method is the pre-assumed linear form of time series. To overcome this limitation, various nonlinear stochastic methods such as Neural Networks have been proposed [8]. The goals of this study are to apply both ARIMA and ANN methods to forecast patients' flow at an hospital emergency department and to compare the accuracy of these methods to the accuracy of the previously proposed forecasting methods for the same problem [1].

2 Forecasting Methods

2.1. Autoregressive Integration Moving Average

Autoregressive integrated moving average method is generally denoted ARIMA (p, d, q) where p is the order of autoregressive process, d is the degree of differencing, and q is the order of the moving average process. To build a time series model using ARIMA, we need to examine the time series data and identify p, d, q. The first step is to examine whether the time series data is stationary or if differencing is required. The second step is to identify the appropriate values of p and q and then estimate the correct ARIMA model. Diagnostic checking is the next step to see if the model is adequate otherwise return to the identification step. Once the ARIMA model has been selected, it will be used to generate forecast [2].

In this study, we developed an automatic algorithm for time series forecasting which determines a suitable time series model, estimate the appropriate parameters and generates forecast. The algorithm uses a variation of the Hyndman and Khandakar algorithm which tries all the possible parameters for a given time series and returns the model with the lowest AIC (Akaike's Information Criterion) and BIC (Bayesian Information Criterion) [11].

2.2. Artificial Neural Networks

Artificial Neural Networks (ANN) are designed to model the human brain based on its neural structure. Like humans' brain, ANN get familiar with the problem to be solved within the training process and are later able to solve the same problems [7]. ANN consist of many interconnected artificial neurons. Neurons are organized in three interconnection layers: input, hidden that may include more than one layer, and output. The input layer receives data from the outside world and represent the inputs to the network. Hidden layers discover relationships between features in the input data by applying non-linear activation functions. The output layer contains the output produced by the network.

Since forecasting is the process of making predictions of future based on identifying patterns in observed time series data, ANN could be used as a forecasting technique [10]. The time series splitted into train and test datasets. Models will be developed using the training dataset and will make predictions on the test dataset. In this study, we apply Recurrent Neural Networks for time series forecasting. A RNN is a class of ANN which has backward connection between hidden layers and is able to process sequential data. A RNN with different activation functions and various number of neurons and hidden layers is developed and tested in this study.

3 Experimental Study

The first objective of the experimental study is to demonstrate the effectiveness of the ARIMA and ANN forecasting methods and to find the optimal neural network structure and ARIMA model parameters. The second objective is to compare the performance of the proposed methods with the performance of the Short-Term and Long-Term methods proposed by Afilal et al. [1]. The techniques used involve linear regression and ARIMA model with fixed and similar parameters for all types of time series.

The arrival data from the ED of a French hospital over the last 7 years are available. To evaluate the developed ARIMA model, we arbitrarily chose the data of May 2017 and November 2017. Experiments are conducted on time series with various number of observations for single-day-ahead and 7-day-ahead forecast. Table 1 represents the performance of the ARIMA model with specified parameters. The results show acceptable performance of the ARIMA model in all configuration of parameters. Its average-case performance is about 92%. However, no clear conclusion about the optimal number of observations and forecast days can be made.

Table 1: ARIMA model performance at May 2015 and November 2017

	May 2017		November 2017	
Time series (Nb. weeks)	n. ahead=1	n. ahead=7	n. ahead=1	n. ahead=7
2	90%	91%	91%	92%
5	91%	**92%**	91%	92%
9	91%	**92%**	**93%**	92%
15	**92%**	**92%**	92%	92%
30	**92%**	**92%**	92%	92%

To evaluate the performance of ANN, first we have to specify amount of training data. There is no simple technique to identify optimal quantity and quality of training data for a problem. In practice, the longer training sample implicitly guarantee that the data contain valuable information of the hydrological behavior. Thus, in this study, the experiments are constructed by using all data available. The data is splitted into the training set from January 2010 to December 2016 and the test set of 2017. To find an optimal configuration of ANN structure, various scenarios are designed by varying the number of neurons, layers and lag (see Table 2). The results show that ANN performs better by increasing the number of layers and lags. However, the accuracy of this algorithm in any cases is less than proposed ARIMA ones.

The Long-Term (LTF) and Short-Term (STF) forecasting methods proposed by Afilal et al. [1] are evaluated on the same data sets. The authors used fixed ARIMA parameters (p, d, q) = (7,0,7). The accuracy of LTF at May 2017 and November 2017 is 90% and 91% respectively. On the same data sets, the accuracy of STF is 89% and 91%. The results shown that developed ARIMA model in this study performs slightly better than others.

Table 2 : ANN performance over 2017

	Artificial Neural Network Models				
	Scenario 1	Scenario 2	Scenario 3	Scenario 4	Scenario 5
Nb. Lag	7	7	3	3	1
Nb. Layers	1	3	1	3	1
Nb. Neurons	7	7	7	7	7
Accuracy	90%	91%	88%	88%	86%

4 Conclusions

The daily arrivals at the emergency department in a French hospital center is studied in this paper. Two forecasting Artificial Neural Networks and ARIMA models are developed. This study focused on automatically identifying the optimal ARIMA model. The proposed models have been evaluated on real-world data and are compared with various forecasting models. The experiments showed that the proposed ARIMA model performed slightly better than other forecasting algorithms with very acceptable average performance: 92%. This is mainly due to the dynamic parameters' settings used in our work. In addition, a number of scenarios are evaluated and compared to determine the suitable configuration of the models. But, it was not possible to conclude about the optimal configuration. Our future work includes estimating the right number of time series observations to balance between computational times and the quality of the results and automatically identifying the optimal parameters of ANN model.

References

[1] Afilal, M., Yalaoui, F., Dugardin, F., Amodeo, L., Laplanche, D., & Blua, P. (2016). Forecasting the emergency department patients flow. Journal of medical systems, 40(7), 175.

[2] Adhikari, R., & Agrawal, R. K. (2013). An introductory study on time series modeling and forecasting. arXiv preprint arXiv:1302.6613

[3] Bowerman, B. L., & O'Connell, R. T. (1986). Time series forecasting: unified concepts and computer implementation.

[4] Box, G. E., & Jenkis, G. M. (1970). Time series analysis for casting and control (No. 519.232 B6).

[5] De Gooijer, J.G., and Hyndman, R.J. (2006). 25 years of time series forecasting. Int. J. Forecast. 22(3):443–473.

[6] Ho, S. L., & Xie, M. (1998). The use of ARIMA models for reliability forecasting and analysis. Computers & industrial engineering, 35(1-2), 213-216.

[7] Jain, A. K., Mao, J., & Mohiuddin, K. M. (1996). Artificial neural networks: A tutorial. Computer, 29(3), 31-44.

[8] Moreno, J. J. M., Pol, A. P., & Gracia, P. M. (2011). Artificial neural networks applied to forecasting time series. Psicothema, 23(2), 322-329.

[9] Tandberg, D., & Qualls, C. (1994). Time series forecasts of emergency department patient volume, length of stay, and acuity. Annals of emergency medicine, 23(2), 299-306.

[10] Zhang, G. (1998). Linear and nonlinear time series forecasting with artificial neural networks. Kent State University.

[11] Hyndman, R. J., & Khandakar, Y. (2007). Automatic time series for forecasting: the forecast package for R (No. 6/07). Monash University, Department of Econometrics and Business Statistics.

Single-objective Real-parameter Optimization:Enhanced LSHADE-SPACMA Algorithm

Anas A. Hadi[1], Ali W. Mohamed [2], and Kamal M. Jambi [1]

[1]*Faculty of Computing and Information Technology, King Abdulaziz University, P. O. Box 80200, Jeddah 21589, Saudi Arabia*

[2]*College of Engineering and Architecture, Al-Yamamah University, P.O. Box 45180, Riyadh 11512, Saudi Arabia*

Abstract

Real parameter optimization is one of the active research fields during the last decade. The performance of LSHADE-SPACMA was competitive in IEEE CEC'2017 competition on Single Objective Bound Constrained Real-Parameter Single Objective Optimization. Besides, it was ranked fourth among twelve papers were presented on and compared to this new benchmark problems. In this work, an improved version named ELSHADE-SPACMA is introduced. In LSHADE-SPACMA, *p* value that controls the greediness of the mutation strategy is constant. While in ELSHADE-SPACMA, *p* value is dynamic. Larger value of *p* will enhance the exploration, while smaller values will enhance the exploitation. We further enhanced the performance of ELSHADE-SPACMA by integrating another directed mutation strategy within the hybridization framework. The proposed algorithm has been evaluated using IEEE CEC'2017 benchmark. According to the comparison results, the proposed ELSHADE-SPACMA algorithm is better than LSHADE and LSHADE-SPACMA. Besides, The comparison results between ELSHADE-SPACMA and the best three algorithms from the IEEE CEC'2017 Competition indicate that ELSHADE-SPACMA algorithm shows overall better performance and it is highly competitive algorithm for solving global optimization problems.

Keywords*:* Evolutionary computation***;***Numerical Optimization; Differential Evolution; LSHADE; Parameter adaptation

Enhanced LSHADE with Semi-Parameter Adaptation Hybrid with CMA-ES (ELSHADE-SPACMA)

In this section, we describe the details of ELSHADE-SPACMA which is a new improvement of LSHADE-SPCMA [1]. A brief background about each component of ELSHADE-SPACMA is given, then, the new improved version will be discussed.

LSHADE Algorithm

LSHADE algorithm proposed by Tanabe and Fukunaga [2]. In order to establish a starting point for the optimization process, an initial population P^0 must be created. Typically, each j^{th} component $(j = 1,2,.....,D)$ of the i^{th} individuals $(i = 1,2,.....,NP)$ in the P^0 is obtained as follow:

) (1)

Where rand (0,1) returns a uniformly distributed random number in [0, 1].

At generation G, for each target vector x_i^G, a mutant vector is generated according to current-to-pbest/1 mutation strategy which was proposed by in the framework of JADE by Zhang and A. C. Sanderson [3].

) (2)

The *P* value here is considered as a control parameter for the greediness of the mutation strategy in order to balance exploitation and exploration. r_1 is a random index selected from the population,r_2 is another random index selected from the concatenation of the population with an external archive. This external archive holds parent vectors which successfully produced better vectors. x_{pbest}^{G} is the best individual vector with the best fitness value in the population at generation *G*. The scale factor $F_i^G$is a positive control parameter for scaling the difference vector.

In the crossover, the target vector is mixed with the mutated vector, using the following scheme, to yield the trial vector u_i^G.

$$u_{i,j}^{G} = \begin{cases} v_{i,j}^{G} \ , if \ (rand_{i,j} \leq CrORj = \ j_{rand}) \\ \qquad x_{i,j}^{G} \ , \ otherwise \end{cases} \qquad (3)$$

Where $rand_{j,i} \ i \in \{1, N\} \ and \ j \ \in \{1 \ , D\}$is a uniformly distributed random number in [0,1], $Cr \in$ [0,1] called the crossover rate that controls how many components are inherited from the mutant vector, j_{rand} is a uniformly distributed random integer in[1, *D*] that makes sure at least one component of trial vector is inherited from the mutant vector.

DE adapts a greedy selection strategy. If and only if the trial vectoru_i^Gyields as good as or a better fitness function value than x_i^G, then $u_i^G$is set to x_i^{G+1}. Otherwise, the old vector $x_i^G$is reserved. The selection scheme is as follows (for a minimization problem):

$$x_i^{G+1} = \begin{cases} u_i^G \ , if \ (f(u_i^G) \leq \ f(x_i^G)) \\ \quad x_i^G \ , \ otherwise \end{cases} \qquad (4)$$

In order to improve the performance of LSHADE-SPA, Linear Population Size Reduction (LPSR) was used. In LPSR the population size will be decreased according to a linear function. The linear function in LSHADE-SPA was:

$$N_{G+1} = round[\left(\frac{N^{min} \_ \ N^{init}}{MAX_{NFE}}\right) * NFE + \ N^{init}] \qquad (5)$$

Where *NFE* is the current number of fitness evaluations, *MAX_NFE* is the maximum number of fitness evaluations, Ninit is the initial population size, and *Nmin* = 4 which is the minimum number of individuals that DE can work with.

CMA-ES Algorithm

Among many variants of Evolution Strategies, CMA-ES was efficiently able to solve diverse types of optimization problems [4]. In CMA-ES the search space is modeled using multivariate normal distribution. New individuals are generated using Gaussian distribution considering the path that the population takes over generations. CMA-ES automatically adapt the mean vector m, covariance matrix C, and step size σ.

CMA-ES steps are as the following:

1. Create an initial population and evaluate the fitness function.
2. Generate new individuals using Gaussian distribution: $x_i = N(m, \sigma^2 C) \ \forall \ i = 1{:}n$ (6)
3. Mean vector m is updated using best μ individuals according to: $m = \sum_{i=1}^{\mu} w_i x_i$ where $\sum_{i=1}^{\mu} w_i = 1$ and $w1 \ \geq w2 \ \geq \ldots \geq \ w\mu.$
4. Step size σ and Covariance matrix C are updated.
5. Repeat steps 2 to 3 until a stopping criterion is met.

For more details about CMA-ES are illustrated in [26].CMA-ES MATLAB code used in this paper was downloaded from [5]. In order to improve the exploration capability of LSHADE-SPACMA, a crossover operation was applied after the CMA-ES offspring generation step according crossover equation Eq.3.

Semi-Parameter adaptation of Scaling Factor (F) and Crossover Rate (Cr)

Parameter setting has a significant impact on the performance of DE. The practices in the fields of parameter adaptation demonstrate the relationship between the problem itself and the parameter values [6]. Each problem has its own appropriate parameter values. In order to perform Semi-Parameter Adaptation (SPA) for F and *Cr*, SPA is composed of two parts. The first part is activated during the first half of the search, while the second part is activated during the second half of the search.

First part of SPA

The idea is to activate the change one parameter at a time policy. Thus, during the first part of SPA, the adaptation is concentrated on one parameter *Cr* using LSHADE adaptation, while *F* parameter will be generated using uniform distribution randomly within a specific limit. The first part of SPA using the condition: (*nfes* < *max_nfes/2*) where *nfes* is the current number of function evaluations and *max_nfes* is the maximum number of function evaluation.

During SPA, each individual has its own and Cr values. will be generated using uniform distribution within the range (0.45,0.55): (7)

On the other hand, values is adapted according to the following equation: (8)

Where is a randomly selected memory slot where successful means of previous generations which are stored. Memory index i is selected randomly from the range [1,h] where h here is the memory size. Initially, all values are set to 0.5, and by the end of each generation, one memory slot is updated using the arithmetic mean of Cr_i values, which successfully generate new individuals.

Second part of SPA

During the second part, L-SHADE adaptation will be used to adapt *F* parameter adaptation using the following equation. The adaptation will be concentrated on *F*. Each individual has its own F_i value. F_i will be generated using Cauchy distribution: $F_i = randc(MF_i, \sigma)$ (9)

is the standard deviation for Cauchy distribution and it was set to 0.1,MF is a randomly selected memory slot where successful means of previous generations which are stored. By the end of each generation, one memory slot is updated using the Lehmer mean of values, which successfully generate new individuals. values of the last 5 generation of the first part of SPA are used to initialize memory slot for the second part of SPA.

Cr parameter adaptation process will remain as is during the second this part. Due to the nature of LSHADE parameter adaptation, *Cr* parameter will be gradually frozen to the adapted values. According to LSHADE parameter adaptation, when all *Cr* values in a generation are failed to generate successful individuals, the corresponding memory slot is set to a terminal value. Thus, *Mcr* will be frozen until the end of the search.

LSHADE-SPACMA Algorithm

LSHADE-SPACMA framework starts with a mutual population *P*. Each individual *x* in *P* will generate offspring individual *u* using either LSHADE or CMA-ES. This assignment is done according to class probability variable (*FCP*). *FCP* values are randomly selected from memory slots M_{FCP}. By the end of each generation, one memory slot M_{FCP} is updated according to the performance of each algorithm. Thus, more populations will be assigned gradually to the better performance algorithm. The update is performed using individuals that successfully generate new individuals only. Memory slot M_{FCP} is updated according to:

(10)

where *c* is the learning rate, and is the improvement rate for each algorithm.

(11)

Where 0.2 and 0.8 values are the minimum and maximum probabilities assigned to each algorithm. Thus, to maintain both algorithms executed simultaneously, FCP values will be always kept in the range (0.2, 0.8). ω_{Alg1} is the summation of differences between old and new fitness values for each individual belongs algorithm $Alg1$.

(12)

Where is the fitness function, is the old individual, is the offspring individual, and is the number of individuals belongs to algorithm 1. Fig.S1 shows LSHADE-SPACMA pseudo code.

AGDE Mutation Strategy

AGDE mutation strategy was proposed in [7].In order to utilize the information of good and bad vectors in the DE population, AGDE integrate information from the best and worst groups from the population. Fig.2 shows AGDE pseudo code. AGDE uses two random chosen vectors of the top and the bottom 100p% individuals in the current population of size *NP* while the third vector is selected randomly from the middle [*NP-2*(100p %)] individuals. This new mutation scheme helps maintain effectively the balance between the global exploration and local exploitation abilities for searching process of the DE. In each generation, the population is divided into three clusters (best, better and worst) of sizes 100p%, NP-2*(100p%) and 100p% respectively. Three vectors are selected randomly, one from each partition to generate the mutant vector based on the following equation:

(13)

Where x_r is chosen randomly from the middle NP-2*(100p%), $x_{p-best}, x_{p-worst}$ are chosen randomly from the top and bottom 100p%, where $F$is the mutation factors that are independently generated according to uniform distribution in [0.1,1]. Fig.S2 shows AGDE-SPA pseudo code.

ELSHADE-SPACMA Hybridization framework

Two improvement were considered to improve the performance of LSHADE-SPACMA. The first one was a hybridization framework between LSHADE-SPACMA and AGDE. Both algorithms were integrated with 50% for each of them. The framework will assign all the population to LSHADE-SPACMA for one generation, then all the population will be assigned to AGDE for another generation. ELSHADE-SPACMA framework is illustrated in Fig.S3.

The second improvement was to balance the exploration and exploitation behavior of ELSHADE-SPACMA by adjusting the greediness parameter p of the mutation. p value will start with a larger value in order to enhance the exploration capability of ELSHADE-SPACMA, then it will be reduced linearly in order to concentrate the search and enhance the exploitative capability of ELSHADE-SPACMA during late stages of the search. p value will be reduced according to:

$$\frac{\quad}{\quad} \qquad] \qquad (14)$$

Where NFE is the current number of fitness evaluations, MAX_NFE is the maximum number of fitness evaluations, is the initial p value, and p^{min} is the minimum value of p.

Experimental study

Numerical benchmarks

The performance of the proposed ELSHADE-SPACMA algorithm is evaluated using a set of problems presented in the CEC2017 competition on real-parameter single objective optimization. This benchmark contains 30 test functions with a diverse set of characteristics. D is the dimensionality of the problem and the functions are tested on 10D, 30D, 50D and 100D. In summary, functions 1-3 are unimodal, functions 4-10 are multimodal, functions 11-20 are hybrid functions and 21-30 are composition functions. More details can be found in [8]. Note that f_2 has been excluded because it shows unstable behavior especially for higher dimensions.

Parameter settings and involved algorithms

Algorithm parameters for ELSHADE-SPACMA are as the following. The initial population size (*NP*) were set to 18*D, Memory size *(H)*, and archive rate (*Arc_rate*) were set to 5 and 1.4 as it was described in LSHADE. Probability Variable (F_{CP}) was set to 0.5, and learning rate (c) was set to 0.8. The threshold, where the second part of SPA is activated, was set to (*max_nfes/2*). AGDE p value were set to 0.1. Finally, p^{init} was set to 0.3 and p^{min} was set to 0.15. ELSHADE-SPACMA is compared with best three algorithms from the IEEE CEC'2017 Competition were, in this order, EBOwithCMAR [9], jSO [10] and LSHADE-cnEpSin [11].

Experimental results and discussions

Results of ELSHADE-SPACMA algorithm

To evaluate the performance of algorithms, experiments were conducted on the test suite. We adopt the solution error measure f(x) −f(x*), where x is the best solution obtained by algorithms in one run and x* is the well-known global optimum of each benchmark function. Error values and standard deviations smaller than 10^{-8} are taken as zero. The maximum number of function evaluations (FEs), the terminal criteria, is set to 10000 × D, all experiments for each function and each algorithm run 51 times independently. The statistical results of the ELSHADE-SPACMA on the benchmarks with 10, 30, 50 and 100 dimensions are summarized in Tables 1-4. It includes the obtained best, worst, median, mean values and the standard deviations of error from the optimum solution of the proposed ELSHADE-SPACMA over 51 runs for all 29 benchmark functions.

Table 6 using score metric between LSHADE-cnEpSin, jSO, EBOwithCMAR and ELSHADE-SPACMA algorithms

Algorithms	Score1	Score2	Score	Rank
ELSHADE-SPACMA	50	50	100	1
EBOwithCMAR	48.92592	49.71731	98.64	2
jSO	48.62186	46.62028	95.24	3
LSHADE-cnEpSin	45.80996	47.88972	93.69	4

TABLE 1 RESULTS OF THE 10D BENCHMARK FUNCTIONS, AVERAGED OVER 51 INDEPENDENT RUNS

Func.	Best	Worst	Median	Mean	Std.
f1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f3	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f4	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f5	9.95E-01	1.09E+01	3.00E+00	3.87E+00	2.02E+00
f6	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f7	1.11E+01	1.97E+01	1.25E+01	1.33E+01	1.75E+00
f8	9.95E-01	1.29E+01	3.98E+00	4.10E+00	2.51E+00
f9	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f10	1.14E+00	2.27E+02	6.53E+00	2.27E+01	4.84E+01
f11	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f12	0.00E+00	1.31E+02	4.16E-01	2.85E+01	5.14E+01
f13	0.00E+00	5.39E+00	4.84E+00	3.57E+00	2.21E+00
f14	0.00E+00	9.95E-01	0.00E+00	7.80E-02	2.70E-01
f15	2.63E-07	5.00E-01	1.44E-01	2.51E-01	2.17E-01
f16	2.04E-02	1.14E+00	5.19E-01	5.62E-01	2.55E-01
f17	1.42E-02	4.35E-01	6.05E-02	1.39E-01	1.44E-01
f18	2.29E-04	2.00E+01	3.27E-01	7.10E-01	2.80E+00
f19	0.00E+00	3.92E-02	1.94E-02	1.55E-02	1.14E-02
f20	0.00E+00	3.12E-01	0.00E+00	1.41E-01	1.57E-01
f21	1.00E+02	2.06E+02	1.00E+02	1.02E+02	1.48E+01
f22	1.00E+02	1.00E+02	1.00E+02	1.00E+02	1.21E-01
f23	3.00E+02	3.10E+02	3.05E+02	3.04E+02	2.30E+00
f24	0.00E+00	3.40E+02	3.34E+02	2.91E+02	9.54E+01
f25	3.98E+02	4.46E+02	3.98E+02	4.13E+02	2.18E+01
f26	3.00E+02	3.00E+02	3.00E+02	3.00E+02	0.00E+00
f27	3.89E+02	3.90E+02	3.90E+02	3.89E+02	1.67E-01
f28	0.00E+00	6.12E+02	3.00E+02	3.25E+02	1.04E+02
f29	2.27E+02	2.36E+02	2.30E+02	2.30E+02	2.26E+00
f30	3.95E+02	4.43E+02	3.95E+02	4.02E+02	1.77E+01

TABLE 2 RESULTS OF THE 30D BENCHMARK FUNCTIONS, AVERAGED OVER 51 INDEPENDENT RUNS

Func.	Best	Worst	Median	Mean	Std.
f1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f3	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f4	5.86E+01	5.86E+01	5.86E+01	5.86E+01	0.00E+00
f5	4.97E+00	3.78E+01	1.79E+01	1.86E+01	8.04E+00
f6	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f7	3.31E+01	4.79E+01	3.81E+01	3.89E+01	3.43E+00
f8	6.12E+00	3.58E+01	1.39E+01	1.61E+01	7.46E+00
f9	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f10	1.07E+03	2.81E+03	1.65E+03	1.70E+03	4.06E+02
f11	0.00E+00	6.40E+01	3.99E+00	7.80E+00	1.42E+01
f12	3.93E+00	5.53E+02	2.48E+02	2.47E+02	1.28E+02
f13	2.98E+00	2.16E+01	1.70E+01	1.57E+01	5.03E+00
f14	2.13E+00	3.10E+01	2.50E+01	2.37E+01	5.25E+00
f15	2.99E-01	4.56E+00	1.46E+00	1.86E+00	1.29E+00
f16	3.57E+00	2.89E+02	2.56E+01	6.68E+01	8.35E+01
f17	1.05E+01	4.13E+01	2.92E+01	2.97E+01	6.76E+00
f18	2.94E-01	2.28E+01	2.14E+01	2.09E+01	3.03E+00
f19	2.67E+00	8.18E+00	4.62E+00	4.61E+00	1.35E+00
f20	1.67E+01	4.25E+01	2.69E+01	2.73E+01	4.56E+00
f21	2.10E+02	2.36E+02	2.22E+02	2.22E+02	6.64E+00
f22	1.00E+02	1.00E+02	1.00E+02	1.00E+02	0.00E+00
f23	3.49E+02	4.00E+02	3.69E+02	3.69E+02	1.05E+01
f24	4.21E+02	4.57E+02	4.42E+02	4.41E+02	7.84E+00
f25	3.87E+02	3.87E+02	3.87E+02	3.87E+02	9.60E-03
f26	8.95E+02	1.27E+03	1.07E+03	1.08E+03	8.68E+01
f27	4.86E+02	5.11E+02	4.98E+02	4.99E+02	6.15E+00
f28	3.00E+02	4.14E+02	3.00E+02	3.02E+02	1.60E+01
f29	3.63E+02	4.58E+02	4.35E+02	4.33E+02	1.56E+01
f30	1.94E+03	2.12E+03	1.97E+03	1.98E+03	3.34E+01

TABLE 3 RESULTS OF THE 50D BENCHMARK FUNCTIONS, AVERAGED OVER 51 INDEPENDENT RUNS

Func.	Best	Worst	Median	Mean	Std.
f1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f3	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f4	8.02E+00	1.42E+02	2.85E+01	4.36E+01	3.62E+01
f5	9.95E-01	2.98E+01	1.39E+01	1.39E+01	5.55E+00
f6	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f7	5.54E+01	7.35E+01	6.12E+01	6.15E+01	3.86E+00
f8	6.96E+00	4.28E+01	1.69E+01	1.79E+01	7.47E+00
f9	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f10	2.69E+03	5.40E+03	3.59E+03	3.69E+03	6.07E+02
f11	2.03E+01	3.31E+01	2.52E+01	2.62E+01	3.76E+00
f12	4.03E+02	2.28E+03	1.39E+03	1.36E+03	3.42E+02
f13	1.07E+01	6.71E+01	4.51E+01	3.68E+01	1.72E+01
f14	2.52E+01	4.25E+01	2.94E+01	3.07E+01	3.95E+00
f15	1.94E+01	2.91E+01	2.22E+01	2.28E+01	2.20E+00
f16	1.29E+02	8.66E+02	3.57E+02	4.15E+02	1.77E+02
f17	3.48E+01	5.09E+02	2.26E+02	2.30E+02	9.68E+01
f18	2.16E+01	3.43E+01	2.47E+01	2.51E+01	2.56E+00
f19	8.46E+00	1.92E+01	1.44E+01	1.44E+01	2.31E+00
f20	3.66E+01	3.01E+02	8.59E+01	1.08E+02	7.31E+01
f21	2.23E+02	2.62E+02	2.41E+02	2.42E+02	9.52E+00
f22	1.00E+02	4.85E+03	1.00E+02	7.16E+02	1.44E+03
f23	4.35E+02	4.89E+02	4.61E+02	4.62E+02	1.39E+01
f24	5.14E+02	5.62E+02	5.33E+02	5.34E+02	9.14E+00
f25	4.77E+02	4.92E+02	4.80E+02	4.81E+02	2.80E+00
f26	1.08E+03	1.64E+03	1.33E+03	1.34E+03	1.38E+02
f27	4.80E+02	5.28E+02	5.09E+02	5.10E+02	9.52E+00
f28	4.59E+02	5.08E+02	4.59E+02	4.60E+02	6.84E+00
f29	3.27E+02	4.10E+02	3.56E+02	3.58E+02	1.78E+01
f30	5.79E+05	6.63E+05	5.90E+05	5.97E+05	2.38E+04

TABLE 4 RESULTS OF THE 100D BENCHMARK FUNCTIONS, AVERAGED OVER 51 INDEPENDENT RUNS

Func.	Best	Worst	Median	Mean	Std.
f1	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f3	0.00E+00	2.81E-06	4.59E-08	1.60E-07	4.00E-07
f4	1.92E+02	2.12E+02	1.97E+02	2.01E+02	8.67E+00
f5	1.09E+01	2.89E+01	1.79E+01	1.78E+01	3.85E+00
f6	0.00E+00	9.37E-08	0.00E+00	0.00E+00	1.34E-08
f7	1.08E+02	1.15E+02	1.11E+02	1.11E+02	1.48E+00
f8	1.19E+01	2.59E+01	1.79E+01	1.82E+01	3.02E+00
f9	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
f10	7.74E+03	1.23E+04	1.08E+04	1.08E+04	9.53E+02
f11	3.46E+01	2.45E+02	5.22E+01	7.34E+01	4.30E+01
f12	3.44E+03	1.87E+04	6.76E+03	7.79E+03	2.92E+03
f13	6.97E+01	2.88E+02	1.49E+02	1.49E+02	3.83E+01
f14	3.69E+01	6.18E+01	4.69E+01	4.75E+01	5.69E+00
f15	5.23E+01	2.22E+02	9.09E+01	1.08E+02	4.34E+01
f16	7.10E+02	3.00E+03	1.79E+03	1.76E+03	4.88E+02
f17	4.81E+02	1.92E+03	1.27E+03	1.27E+03	3.45E+02
f18	5.62E+01	2.02E+02	1.02E+02	1.05E+02	2.56E+01
f19	4.25E+01	7.45E+01	6.09E+01	6.05E+01	7.55E+00
f20	4.88E+02	1.46E+03	8.89E+02	9.28E+02	2.54E+02
f21	2.64E+02	3.33E+02	2.95E+02	2.96E+02	1.65E+01
f22	7.95E+03	1.20E+04	9.29E+03	9.70E+03	1.20E+03
f23	5.53E+02	6.57E+02	6.01E+02	6.03E+02	2.19E+01
f24	8.91E+02	9.78E+02	9.32E+02	9.32E+02	1.90E+01
f25	6.37E+02	7.74E+02	6.98E+02	7.00E+02	3.99E+01
f26	2.98E+03	4.12E+03	3.17E+03	3.24E+03	2.19E+02
f27	5.30E+02	6.08E+02	5.62E+02	5.62E+02	1.75E+01
f28	4.78E+02	5.77E+02	5.22E+02	5.21E+02	2.38E+01
f29	8.04E+02	1.77E+03	1.21E+03	1.21E+03	1.98E+02
f30	2.09E+03	2.61E+03	2.22E+03	2.25E+03	1.11E+02

Table 5 Comparison between EBOwithCMAR(EBO), jSO , LSHADE-cnEpSin(EpSin),ELSHADE-SPACMA(ESPACMA) on the benchmark with 10,30,50 and 100 dimensions

	D=10				**D=30**				**D=50**				**D=100**			
	EBO	**jSO**	**EpSin**	**ESPACMA**	**EBO**	**jSO**	**EpSin**	**ESPACMA**	**EBO**	**jSO**	**EpSin**	**ESPACMA**	**EBO**	**jSO**	**EpSin**	**ESPACMA**
F1	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F3	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	2.99E-07	2.39E-06	**0.00E+00**	1.60E-07
F4	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	5.65E+01	5.87E+01	**4.23E+01**	5.86E+01	**4.29E+01**	5.62E+01	5.14E+01	4.36E+01	1.93E+02	**1.90E+02**	1.98E+02	2.01E+02
F5	0.00E+00	1.76E+00	**1.69E+00**	3.87E+00	**2.78E+00**	8.56E+00	1.23E+01	1.86E+01	**7.58E+00**	1.64E+01	2.52E+01	1.39E+01	2.87E+01	4.39E+01	5.59E+01	**1.78E+01**
F6	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	8.54E-08	1.09E-06	9.16E-07	**0.00E+00**	1.63E-05	2.02E-04	6.02E-05	**0.00E+00**
F7	**1.06E+01**	1.18E+01	1.20E+01	1.33E+01	**3.35E+01**	3.89E+01	4.33E+01	3.89E+01	**5.79E+01**	6.65E+01	7.66E+01	6.15E+01	1.22E+02	1.45E+02	1.62E+02	**1.11E+02**
F8	**0.00E+00**	1.95E+00	1.80E+00	4.10E+00	2.02E+00	9.09E+00	1.29E+01	1.61E+01	**7.91E+00**	1.70E+01	2.63E+01	1.79E+01	2.97E+01	4.22E+01	5.35E+01	**1.82E+01**
F9	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	1.76E-03	4.59E-02	**0.00E+00**	**0.00E+00**
F10	3.72E+01	3.59E+01	4.30E+01	**2.27E+01**	1.41E+03	1.53E+03	**1.39E+03**	1.70E+03	**3.11E+03**	3.14E+03	3.20E+03	3.69E+03	9.91E+03	**9.70E+03**	1.03E+04	1.08E+04
F11	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	4.49E+00	**3.04E+00**	1.35E+01	7.80E+00	2.64E+01	2.79E+01	**2.14E+01**	2.62E+01	6.56E+01	1.13E+02	**4.92E+01**	7.34E+01
F12	9.02E+01	**2.66E+00**	1.01E+02	2.85E+01	4.63E+02	**1.70E+02**	3.72E+02	2.47E+02	1.94E+03	1.68E+03	1.48E+03	**1.36E+03**	4.19E+03	1.84E+04	4.62E+03	**7.79E+03**
F13	**2.17E+00**	2.96E+00	3.66E+00	3.57E+00	1.49E+01	**1.48E+01**	1.73E+01	1.57E+01	4.14E+01	**3.06E+01**	6.94E+01	3.68E+01	2.45E+02	1.45E+02	**1.25E+02**	1.49E+02
F14	6.05E-02	**5.85E-02**	7.80E-02	7.80E-02	2.19E+01	2.18E+01	**2.16E+01**	2.37E+01	3.12E+01	**2.50E+01**	2.65E+01	3.07E+01	1.38E+02	6.43E+01	4.97E+01	**4.75E+01**
F15	**1.09E-01**	2.21E-01	3.24E-01	2.51E-01	3.69E+00	**1.09E+00**	3.24E+00	1.86E+00	2.94E+01	2.39E+01	2.56E+01	**2.28E+01**	1.65E+02	1.62E+02	**8.99E+01**	1.08E+02
F16	**4.17E-01**	5.69E-01	5.37E-01	5.62E-01	4.26E+01	7.89E+01	**2.29E+01**	6.68E+01	3.46E+02	4.51E+02	**2.75E+02**	4.15E+02	1.41E+03	1.86E+03	**1.22E+03**	1.76E+03
F17	1.47E-01	5.02E-01	3.07E-01	**1.39E-01**	2.98E+01	3.29E+01	**2.86E+01**	2.97E+01	2.75E+02	2.83E+02	**2.07E+02**	2.30E+02	1.21E+03	1.28E+03	**9.32E+02**	1.27E+03
F18	7.00E-01	**3.08E-01**	3.86E+00	7.10E-01	2.21E+01	**2.04E+01**	2.11E+01	2.09E+01	3.20E+01	**2.43E+01**	**2.43E+01**	2.51E+01	2.37E+02	1.67E+02	**7.79E+01**	1.05E+02
F19	1.50E-02	**1.07E-02**	4.47E-02	1.55E-02	8.04E+00	**4.50E+00**	5.83E+00	4.61E+00	2.45E+01	**1.41E+01**	1.74E+01	1.44E+01	1.15E+02	1.05E+02	**5.55E+01**	6.05E+01
F20	1.47E-01	3.43E-01	2.57E-01	**1.41E-01**	3.57E+01	2.94E+01	3.03E+01	**2.73E+01**	1.47E+02	1.40E+02	1.14E+02	**1.08E+02**	1.36E+03	1.38E+03	1.08E+03	**9.28E+02**
F21	1.14E+02	1.32E+02	1.46E+02	**1.02E+02**	**1.99E+02**	2.09E+02	2.12E+02	2.22E+02	**2.11E+02**	2.19E+02	2.27E+02	2.42E+02	**2.60E+02**	2.64E+02	2.77E+02	2.96E+02
F22	**9.85E+01**	1.00E+02	1.00E+02	1.00E+02	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**3.65E+02**	1.49E+03	1.59E+03	7.16E+02	1.02E+04	1.02E+04	1.04E+04	**9.70E+03**
F23	**3.00E+02**	3.01E+02	3.02E+02	3.04E+02	**3.51E+02**	**3.51E+02**	3.56E+02	3.69E+02	4.34E+02	**4.30E+02**	4.39E+02	4.62E+02	5.77E+02	**5.71E+02**	5.98E+02	6.03E+02
F24	**1.66E+02**	2.97E+02	3.16E+02	2.91E+02	**4.18E+02**	4.26E+02	4.28E+02	4.41E+02	**5.06E+02**	5.07E+02	5.13E+02	5.34E+02	9.19E+02	**9.02E+02**	9.17E+02	9.32E+02
F25	4.12E+02	**4.06E+02**	4.26E+02	4.13E+02	**3.87E+02**	**3.87E+02**	**3.87E+02**	**3.87E+02**	4.89E+02	4.81E+02	**4.80E+02**	4.81E+02	7.16E+02	7.36E+02	**6.84E+02**	7.00E+02
F26	**2.65E+02**	3.00E+02	3.00E+02	3.00E+02	**5.37E+02**	9.20E+02	9.49E+02	1.08E+03	**7.06E+02**	1.13E+03	1.20E+03	1.34E+03	**2.77E+03**	3.27E+03	3.11E+03	3.24E+03
F27	3.92E+02	**3.89E+02**	**3.89E+02**	**3.89E+02**	5.02E+02	**4.97E+02**	5.04E+02	4.99E+02	5.22E+02	5.11E+02	5.25E+02	**5.10E+02**	5.88E+02	5.85E+02	5.89E+02	**5.62E+02**
F28	**3.07E+02**	3.39E+02	3.85E+02	3.25E+02	3.08E+02	3.09E+02	3.15E+02	**3.02E+02**	4.67E+02	4.60E+02	**4.59E+02**	4.60E+02	**5.10E+02**	5.27E+02	5.15E+02	5.21E+02
F29	2.31E+02	2.34E+02	**2.28E+02**	2.30E+02	**4.33E+02**	4.34E+02	4.35E+02	**4.33E+02**	**3.47E+02**	3.63E+02	3.53E+02	3.58E+02	1.28E+03	1.26E+03	**1.12E+03**	1.21E+03
F30	4.07E+02	**3.95E+02**	1.76E+04	4.02E+02	1.99E+03	**1.97E+03**	1.98E+03	1.98E+03	6.18E+05	6.01E+05	6.58E+05	**5.97E+05**	2.40E+03	2.33E+03	2.36E+03	**2.25E+03**

Table 7 Wilcoxon's test between ELSHADE-SPACMA, EBOwithCMAR, jSO and LSHADE-cnEpSin algorithms for D=10, 30 ,50 and 100, respectively

D	Algorithms	R^+	R^-	p-value	+	≈	-	Dec.
10	ELSHADE-SPACMA vs EBOwithCMAR	102.5	173.5	0.280	8	6	15	≈
	ELSHADE-SPACMA vs jSO	95	115	0.709	8	9	12	≈
	ELSHADE-SPACMA vs LSHADE-cnEpSin	144	46	**0.049**	13	10	6	+
30	ELSHADE-SPACMA vs EBOwithCMAR	81	172	0.140	9	7	13	≈
	ELSHADE-SPACMA vs jSO	61	215	**0.019**	7	6	16	+
	ELSHADE-SPACMA vs LSHADE-cnEpSin	96	157	0.322	11	7	11	≈
50	ELSHADE-SPACMA vs EBOwithCMAR	166.5	184.5	0.819	14	3	12	≈
	ELSHADE-SPACMA vs jSO	180.5	119.5	0.383	14	5	10	≈
	ELSHADE-SPACMA vs LSHADE-cnEpSin	180.5	170.5	0.899	13	3	13	≈
100	ELSHADE-SPACMA vs EBOwithCMAR	251	184	0.469	18	0	11	≈
	ELSHADE-SPACMA vs jSO	326.5	79.5	**0.005**	22	1	6	+
	ELSHADE-SPACMA vs LSHADE-cnEpSin	126	225	0.209	8	3	18	≈

Table 9 Results using score metric between LSHADE, LSHADE-SPA, LSHADE-SPACMA and ELSHADE-SPACMA algorithms

Algorithms	Score1	Score2	Score	Rank
ELSHADE-SPACMA	50	46.23	96.23	1
LSHADE-SPACMA	45.44	50	95.44	2
LSHADE-SPA	44.69	39.23	83.92	3
LSHADE	42.53	35.73	78.26	4

Table 10 Wilcoxon's test between LSHADE, LSHADE-SPA, LSHADE-SPACMA and ELSHADE-SPACMA, algorithms for D=10, 30, 50 and 100, respectively

D	Algorithms	R^+	R^-	p-value	+	≈	-	Dec.
10	ELSHADE-SPACMA vs LSHADE-SPACMA	124	107	0.767	13	8	8	≈
	ELSHADE-SPACMA vs LSHADE-SPA	159	94	0.767	12	7	10	≈
	ELSHADE-SPACMA vs LSHADE	120	90	0.575	9	9	11	≈
30	ELSHADE-SPACMA vs LSHADE-SPACMA	114.5	138.5	0.697	11	7	11	≈
	ELSHADE-SPACMA vs LSHADE-SPA	119	134	0.808	10	7	12	≈
	ELSHADE-SPACMA vs LSHADE	134.5	141.5	0.915	12	6	11	≈
50	ELSHADE-SPACMA vs LSHADE-SPACMA	149	127	0.738	12	6	11	≈
	ELSHADE-SPACMA vs LSHADE-SPA	226.5	124.5	0.195	19	3	7	≈
	ELSHADE-SPACMA vs LSHADE	199.5	125.5	0.319	16	4	9	≈
100	ELSHADE-SPACMA vs LSHADE-SPACMA	145	180	0.638	10	4	15	≈
	ELSHADE-SPACMA vs LSHADE-SPA	258	93	**0.036**	17	3	9	+
	ELSHADE-SPACMA vs LSHADE	307.5	98.5	**0.017**	21	1	7	+

From the above results, comparisons and discussion through this section, the proposed ELSHADE-SPACMA algorithm is of better searching quality, efficiency and robustness for solving unconstrained global optimization problems. It is clear that the proposed ELSHADE-SPACMA algorithm performs well and it has shown its outstanding superiority with separable, non-separable, unimodal and multimodal functions with shifts in dimensionality, rotation, multiplicative noise in fitness and composition of functions. Consequently, its performance is not influenced by all these obstacles. Contrarily, it greatly keeps the balance the local optimization speed and the global optimization diversity in challenging optimization environment with invariant performance. Besides, its performances is superior and competitive with the performance of the best three algorithms from the IEEE CEC'2017 Competition. Finally, It can be concluded that the new directed mutation scheme of AGDE and dynamic *p* value help to maintain effectively the balance between the global exploration and local exploitation abilities for searching process of the LSHADE-spacma which enhances significantly its performance during the search process.

Table 8 Comparison between LSHADE, LSHADESPA(SPA), LSHADE-SPACMA(SPACMA ,ELSHADE-SPACMA(ESPACMA) on the benchmark with 10,30,50 and 100 dimensions

	D=10				**D=30**				**D=50**				**D=100**			
	LSHADE	**SPA**	**SPACMA**	**ESPACMA**	**LSHADE**	**SPA**	**SPACMA**	**ESPACMA**	**LSHADE**	**SPA**	**SPACMA**	**ESPACMA**	**LSHADE**	**SPA**	**SPACMA**	**ESPACMA**
F1	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F3	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	2.88E-06	**0.00E+00**	**0.00E+00**	1.60E-07
F4	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**5.86E+01**	**5.86E+01**	**5.86E+01**	**5.86E+01**	6.30E+01	4.96E+01	**2.94E+01**	4.36E+01	**1.96E+02**	2.03E+02	2.01E+02	2.01E+02
F5	2.99E+00	**1.76E+00**	**1.76E+00**	3.87E+00	6.70E+00	1.25E+01	**3.45E+00**	1.86E+01	1.18E+01	2.88E+01	**5.99E+00**	1.39E+01	2.83E+01	5.19E+01	**1.22E+01**	1.78E+01
F6	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	1.47E-08	**0.00E+00**	**0.00E+00**	**0.00E+00**	7.61E-08	2.65E-07	**0.00E+00**	**0.00E+00**	1.66E-03	2.60E-05	**0.00E+00**	**0.00E+00**
F7	1.22E+01	1.19E+01	**1.09E+01**	1.33E+01	3.74E+01	4.30E+01	**3.38E+01**	3.89E+01	6.43E+01	8.01E+01	**5.70E+01**	6.15E+01	1.36E+02	1.57E+02	1.12E+02	**1.11E+02**
F8	2.42E+00	1.87E+00	**8.42E-01**	4.10E+00	7.97E+00	1.27E+01	**3.20E+00**	1.61E+01	1.15E+01	2.89E+01	**5.81E+00**	1.79E+01	2.72E+01	5.02E+01	**1.02E+01**	1.82E+01
F9	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**1.07E-01**	**0.00E+00**	**0.00E+00**	**0.00E+00**
F10	2.23E+01	**2.17E+01**	2.18E+01	2.27E+01	1.40E+03	**1.33E+03**	1.44E+03	1.70E+03	3.09E+03	**2.96E+03**	3.49E+03	3.69E+03	1.02E+04	**9.88E+03**	1.00E+04	1.08E+04
F11	4.15E-01	**0.00E+00**	**0.00E+00**	**0.00E+00**	3.41E+01	1.55E+01	1.78E+01	**7.80E+00**	5.01E+01	2.74E+01	3.22E+01	**2.62E+01**	3.78E+02	**4.38E+01**	5.20E+01	7.34E+01
F12	7.69E+01	1.20E+02	1.19E+02	**2.85E+01**	1.00E+03	4.08E+02	6.15E+02	**2.47E+02**	2.23E+03	1.42E+03	1.56E+03	**1.36E+03**	2.35E+04	5.74E+03	4.80E+03	**7.79E+03**
F13	**3.16E+00**	3.62E+00	4.37E+00	3.57E+00	1.56E+01	1.52E+01	**1.46E+01**	1.57E+01	6.37E+01	4.90E+01	3.71E+01	**3.68E+01**	1.22E+03	**9.42E+01**	1.48E+02	1.49E+02
F14	1.74E-01	**2.04E-02**	1.56E-01	7.80E-02	**2.17E+01**	2.25E+01	2.34E+01	2.37E+01	3.20E+01	**2.74E+01**	2.94E+01	3.07E+01	2.68E+02	5.44E+01	7.14E+01	**4.75E+01**
F15	**1.70E-01**	2.68E-01	4.08E-01	2.51E-01	3.80E+00	2.17E+00	4.46E+00	**1.86E+00**	4.46E+01	2.44E+01	3.04E+01	**2.28E+01**	2.58E+02	**9.71E+01**	1.08E+02	1.08E+02
F16	**4.09E-01**	5.25E-01	7.42E-01	5.62E-01	4.18E+01	3.05E+01	**2.52E+01**	6.68E+01	3.80E+02	**3.00E+02**	3.35E+02	4.15E+02	1.56E+03	1.37E+03	**1.25E+03**	1.76E+03
F17	1.72E-01	**1.19E-01**	1.56E-01	1.39E-01	3.27E+01	**2.85E+01**	3.04E+01	2.97E+01	2.45E+02	2.33E+02	2.77E+02	**2.30E+02**	1.10E+03	1.10E+03	**1.03E+03**	1.27E+03
F18	**2.78E-01**	2.43E+00	4.35E+00	7.10E-01	2.32E+01	2.11E+01	2.34E+01	**2.09E+01**	4.79E+01	**2.49E+01**	3.24E+01	2.51E+01	2.07E+02	**9.75E+01**	1.35E+02	1.05E+02
F19	**1.11E-02**	5.51E-02	2.34E-01	1.55E-02	6.14E+00	4.91E+00	1.03E+01	**4.61E+00**	3.44E+01	1.69E+01	2.17E+01	**1.44E+01**	1.80E+02	**5.85E+01**	7.35E+01	6.05E+01
F20	1.49E-02	1.78E-01	3.12E-01	**1.41E-01**	3.06E+01	2.77E+01	8.38E+01	**2.73E+01**	1.67E+02	1.30E+02	1.68E+02	**1.08E+02**	1.40E+03	1.37E+03	1.47E+03	**9.28E+02**
F21	1.61E+02	1.56E+02	**1.01E+02**	1.02E+02	2.08E+02	2.13E+02	**2.07E+02**	2.22E+02	**2.15E+02**	2.30E+02	**2.15E+02**	2.42E+02	2.57E+02	2.74E+02	**2.44E+02**	2.96E+02
F22	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	2.80E+03	1.53E+03	1.38E+03	**7.16E+02**	1.09E+04	1.00E+04	9.99E+03	**9.70E+03**
F23	3.04E+02	**3.02E+02**	3.03E+02	3.04E+02	**3.55E+02**	**3.55E+02**	**3.55E+02**	3.69E+02	**4.31E+02**	4.41E+02	4.41E+02	4.62E+02	**5.61E+02**	5.90E+02	5.81E+02	6.03E+02
F24	3.21E+02	2.90E+02	**2.75E+02**	2.91E+02	**4.28E+02**	4.29E+02	4.29E+02	4.41E+02	**5.10E+02**	5.14E+02	5.13E+02	5.34E+02	**9.18E+02**	9.32E+02	9.19E+02	9.32E+02
F25	**4.11E+02**	4.25E+02	4.28E+02	4.13E+02	**3.87E+02**	**3.87E+02**	**3.87E+02**	**3.87E+02**	**4.81E+02**	4.82E+02	**4.81E+02**	**4.81E+02**	7.48E+02	**6.91E+02**	7.09E+02	7.00E+02
F26	**3.00E+02**	**3.00E+02**	**3.00E+02**	**3.00E+02**	9.84E+02	9.62E+02	**9.53E+02**	1.08E+03	1.19E+03	1.26E+03	**1.14E+03**	1.34E+03	3.41E+03	3.22E+03	**3.14E+03**	3.24E+03
F27	**3.89E+02**	3.90E+02	3.90E+02	**3.89E+02**	5.08E+02	5.05E+02	5.05E+02	**4.99E+02**	5.41E+02	5.27E+02	5.32E+02	**5.10E+02**	6.62E+02	5.89E+02	5.93E+02	**5.62E+02**
F28	3.59E+02	4.02E+02	**3.17E+02**	3.25E+02	3.42E+02	3.21E+02	3.11E+02	**3.02E+02**	4.63E+02	4.61E+02	**4.60E+02**	**4.60E+02**	5.28E+02	**5.13E+02**	5.17E+02	5.21E+02
F29	2.34E+02	2.31E+02	2.31E+02	**2.30E+02**	4.35E+02	**4.30E+02**	4.45E+02	4.33E+02	3.51E+02	**3.47E+02**	3.92E+02	3.58E+02	1.37E+03	**1.18E+03**	1.59E+03	1.21E+03
F30	7.82E+04	4.09E+04	4.30E+02	**4.02E+02**	2.00E+03	2.00E+03	2.01E+03	**1.98E+03**	6.61E+05	6.24E+05	6.68E+05	**5.97E+05**	2.41E+03	2.39E+03	2.38E+03	**2.25E+03**

Conclusion

In order to enhance the overall performance of ELSHADE-SPACMA algorithm, an improved version named ELSHADE-SPACMA is introduced. In LSHADE-SPACMA, p value that controls the greediness of the mutation strategy is constant. In ELSHADE-SPACMA, p value is dynamic. Larger value of p will enhance the exploration, while smaller values will enhance the exploitation. We further enhanced the performance of ELSHADE-SPACMA by integrating another directed mutation strategy within the hybridization framework. The proposed algorithms were tested on the benchmarks of the CEC2017 which is used in the special Session and Competition on Real-Parameter Single Objective Optimization of the IEEE CEC2017. As a summary of results, the performance of the ELSHADE-SPACMA algorithm was superior to and competitive with LSHADE-SPACMA, LSHADE-SPA and LSHADE algorithms in the majority of functions and for different dimensions especially for high dimension functions with different types. When compared with the best three algorithms from the IEEE CEC'2017 Competition, it shows a very competitive performance and it is ranked first. Moreover, future research will investigate the performance of the ELSHADE-SPACMA algorithm in solving constrained and multi-objective optimization problems as well as real-world applications such as big data, data mining and clustering problems. The MATLAB source code of ELSHADE-SPACMA is available upon request.

References

[1] A. W. Mohamed, A. A. Hadi, A. M. Fattouh, and K. M. Jambi (2017). LSHADE with semi-parameter adaptation hybrid with CMA-ES for solving CEC 2017 benchmark problems. In Evolutionary Computation (CEC), 2017 IEEE Congress on, pp. 145-152.

[2] R.Tanabe, A.S.Fukunaga (2014). Improving the search performance of shade using linear population size reduction. In Proceedings of IEEE Congress on Evolutionary Computation 2014, July 6–11, Beijing, China, pp.1658–1665

[3] J. Zhang, and A. C. Sanderson (2009). JADE: adaptive differential evolution with optional external archive. IEEE Transactions on Evolutionary Computation, 13(5): 945-958.

[4] N. Hansen (2006). The CMA evolution strategy: a comparing review.In Towards a new evolutionary computation: advances on estimation of distribution algorithms, J. A. Lozano, Ed. Springer, 2006, pp.75–102.

[5] N.,Hansen, and A., Ostermeier (2009). Cma-es source code [Online]. Available: http://www.lri.fr/~hansen/cmaes_inmatlab.html

[6] S. Das, S. S. Mullick, and P. N. Suganthan (2016). Recent advances in differential evolution-an updated survey. Swarm and Evolutionary Computation Vol. 27, 2, pp.1–30.

[7] A.W. Mohamed, A.K. Mohamed, "Adaptive guided differential evolution algorithm with novel mutation for numerical optimization." International Journal of Machine Learning and Cybernetics (2017), pp. 1-25, https://doi.org/10.1007/s13042-017-0711-7

[8] N. H. Awad, M. Z. Ali, J. J. Liang, B. Y. Qu and P. N. Suganthan (2016). Problem Definitions and Evaluation Criteria for the CEC 2017 Special Session and Competition on Single Objective Bound Constrained Real-Parameter Numerical Optimization. Technical Report, Nanyang Technological University, Singapore.

[9] A. Kumar, R. K. Misra and D. Singh, "Improving the local search capability of Effective Butterfly Optimizer using Covariance Matrix Adapted Retreat Phase," *2017 IEEE Congress on Evolutionary Computation (CEC)*, San Sebastian, 2017, pp. 1835-1842.

[10] J. Brest, M. S. Maučec and B. Bošković, "Single objective real-parameter optimization: Algorithm jSO," *2017 IEEE Congress on Evolutionary Computation (CEC)*, San Sebastian, 2017, pp. 1311-1318.

[11] N. H. Awad, M. Z. Ali and P. N. Suganthan, "Ensemble sinusoidal differential covariance matrix adaptation with Euclidean neighborhood for solving CEC2017 benchmark problems," *2017 IEEE Congress on Evolutionary Computation (CEC)*, San Sebastian, 2017, pp. 372-379.

Wifi emitters deployment for fingerprinting localization using genetic algorithm

F. Mourad-Chehade[1], H. Chehade[2,3], T. Arbaoui[2] and F. Yalaoui[2]

[1] ICD / M2S, Université de Technologie de Troyes, 10000 Troyes, France
farah.chehade@utt.fr
[2] ICD / LOSI, Université de Technologie de Troyes, 10000 Troyes, France
hicham.chehade@utt.fr,taha.arbaoui@utt.fr,farouk.yalaoui@utt.fr
[3] Opta LP, 10430 Rosières-près-Troyes, France
chehade@opta-lp.fr

1 Introduction

Localization problem is a fundamental issue in wireless sensor networks [1, 2]. This paper deals with the zoning-based localization in indoor WiFi networks. Zoning problems aim to find the zones where the sensors reside in indoor environments [3]. Such an issue is important in many health-care, industrial or environmental applications, that need to find target sensors without analyzing their activity [4, 5]. To localize themselves, sensors collect WiFi signals, measure their strengths then use fingerprinting technology. WiFi signals are mainly emitted by Access Points (APs), already fixed in the environment guaranteeing coverage and thus access to Internet. However, in most of the cases, the APs are not enough to ensure high localization performances.

This paper aims to develop an optimization method, based on genetic algorithm, for placement of additional WiFi emitters in the network to improve at best the localization accuracy, while minimizing the number of added emitters. The additional WiFi emitters are not APs, but low-cost antennas that are not connected to internet. Having a 3D mesh grid over placement possibilities, the problem is an assignment problem that aims at finding the best matching between emitters and grid positions while minimizing the localization error. It is also a dimensioning problem aiming at reducing the number of added emitters. The studied problem belongs to the family of the Set Covering Problem (SCP) which is NP-hard [6, 7]. Moreover, our final goal is to develop an industrial application with low computational time efforts for real world applications with a large number of possible emitters placements. For that reasons, our choice has been oriented for the application of a metaheuristic. Genetic algorithms (GA) have proven their efficiency to solve hard combinatorial optimization problems in different fields. It is a popular and well-known metaheuristic which is often implemented as it can be easily adapted to the studied problem. GA have also been applied to solve the coverage problem in different research works [8, 9].

Many sensors placement methods exist in literature [10, 11]. While most of existing works focus only on the coverage problem, other works consider also localization performances such as in [12]. The novelty of the proposed method, compared to existing ones, remains in considering only the issue of localization accuracy, without dealing with the coverage constraints. Indeed, it assumes that the area of interest has already its WiFi network, that needs to be completed with low-cost emitters to improve the positioning efficiency. This description is closer to reality since most of the indoor environments have already their WiFi equipments. Another originality of this paper is that it considers a zoning-based localization method with exact models and computations. It overcomes the problem of non-circular wave fields, by using the values of signal strengths and matching them to a database constructed by a fingerprinting technique. It performs a novel evaluation of the objective function that works well in real experiments, which is not the case in most of existing works.

2 Localization problem

This paper considers a zoning-based localization, that estimates in real-time the zone where resides a target sensor within an area of interest, denoted by $\mathcal{A}$. The area $\mathcal{A}$ is an indoor environment, structured to zones as illustrated in Fig. 1. Zones could be offices, rooms, halls, corridors identified according to the architecture of $\mathcal{A}$. Let Z_j, $j \in \{1, ..., N_Z\}$, denote the possible zones, A_i, $i \in \{1, ..., N_A\}$, denote the existing APs in the network, fixed at the ceiling of $\mathcal{A}$, and T denote the target sensor to be localized. The objective of the method is then to find the best placement of a minimum number of WiFi emitters, at the walls or the ceiling of $\mathcal{A}$, to improve the localization accuracy. To do this, an optimization algorithm is developed in the following section.

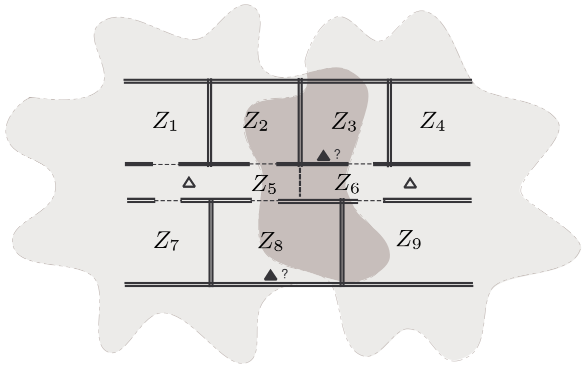


Fig. 1. An example of an indoor area of interest, having 9 zones, 2 APs, denoted by △, and 2 additional WiFi emitters, denoted by ▲. The non-circular WiFi coverage fields of the APs are shown in light gray, with their intersection area in dark gray.

Let E_k, $k \in \{1, ..., N_E\}$, denote the added emitters of number N_E. According to this assumption, a target sensor localizes itself by collecting WiFi signals from the APs and the added emitters. The strengths of these signals decrease with the increase of their traveled distance. For this reason, the proposed method uses the Received Signal Strength Indicators (RSSIs) for localization [3]. It is a fingerprinting-based method, consisting of two phases. At the offline phase, a sensor node equipped with a WiFi receiver moves randomly within each zone of the area of interest $\mathcal{A}$. It collects signals from all APs and emitters and measures their RSSIs. This leads to an offline database that assigns a set of RSSIs to each zone Z_j of $\mathcal{A}$. This database provides a radio-cartography of $\mathcal{A}$, with each zone having a specific signature. Then, for each antenna o and every zone j, the RSSIs are fitted to one of the classical known distributions, denoted by $Q_{j,o}(\cdot)$. Afterwards, using the Belief Functions Theory (BFT) [13], mass functions are defined using the fitted distributions as follows,

$$m_o(Z_j, \cdot) = \frac{Q_{j,o}(\cdot)}{\sum_{j'=1_{Z}^{N}} Q_{j',o}(\cdot)}, \tag{1}$$

where $\cdot$ denotes a measured RSSI from o.

In the online phase, the target sensor T collects some signals from APs and other emitters and measures their RSSIs. Let Θ be the indices of the antennas whose signals are detected by T. Θ depends on the communication range of the emitters and the position of T. Assume $\rho_{T,o}$, $o \in \Theta$, are the corresponding measured RSSIs. Then, the masses $m_o(Z_j, \rho_{T,o})$ are computed using (1). The mass $m_o(Z_j, \rho_{T,o})$ for a given o represents the part of evidence saying that T resides in zone Z_j according to antenna o. Let $\boldsymbol{\rho_T}$ be the vector of elements $\rho_{T,o}$ completed by zeros for missing RSSIs. To combine all masses, the conjunctive combination rule of the BFT is employed as follows,

$$\boldsymbol{m}(Z_j, \boldsymbol{\rho_T}) = \prod_{o \in \Theta} m_o(Z_j, \rho_{T,o}).$$

The target's zone is finally estimated by the one having the highest mass $\hat{Z}_T = \text{argmax}_{Z_j} \boldsymbol{m}(Z_j, \boldsymbol{\rho_T})$.

3 Wifi emitters placement

In order to optimally place the WiFi emitters, we develop an optimization method based on Genetic Algorithm (GA). The GA developed in this work is based on an efficient and innovative computation of the objective function mainly for the accuracy measurement. To have a discrete placement problem, we generate a mesh grid over the walls and the ceiling of the area of interest $\mathcal{A}$, with dimensions adapted as needed. This leads to a huge set of candidate grid positions for the emitters. The objective is then to find the best positions within this set for the emitters,

0	1	1	0	1	0	···	0

Fig. 2. An example of GA solution encoding.

while minimizing their number, to maximize the localization accuracy and thus to minimize the localization error.

The first step of the GA is the solution encoding. For that, we have adopted the solution encoding illustrated in Fig. 2. The number of the genes in the chromosome (length of the chromosome) is equal to the number of the candidate grid positions. We have also adopted a binary encoding. The value of each gene is then a binary variable: 0 means that there is no emitter on the position and 1 otherwise. For each solution $\mathcal{S}$ of GA, the objective function $f(\mathcal{S})$ is evaluated. It consists of a weighted combination of the localization error $\mathcal{E}(\mathcal{S})$ and the normalized number $\mathcal{N}(\mathcal{S})$ of added emitters, both to be minimized,

$$f(\mathcal{S}) = \alpha\mathcal{E}(\mathcal{S}) + (1-\alpha)\mathcal{N}(\mathcal{S}), \tag{2}$$

$\alpha \in [0,1]$ being a tuning parameter, measuring the tradeoff between both quantities. To evaluate this error $\mathcal{E}(\mathcal{S})$, we generate N_p random positions within the area of interest $\mathcal{A}$, denoted by $\boldsymbol{p_\ell}$. A sensor is placed at each position $\boldsymbol{p_\ell}$, measures the RSSIs of APs and of the emitters of the solution $\mathcal{S}$. Its zone is then estimated using the localization algorithm of Section 2. Let $\mathbb{I}_\ell$ be the indicator of the accuracy of the estimation, that is, $\mathbb{I}_\ell = 1$ if the estimate is correct and 0 otherwise. Then,

$$\mathcal{E}(\mathcal{S}) = 1 - \frac{1}{N_p}\sum_{\ell=1}^{N_p}\mathbb{I}_\ell. \tag{3}$$

$\mathcal{E}(\mathcal{S})$ represents the percentage of the erroneous estimations for the solution $\mathcal{S}$. The number of added emitters is normalized to have a value in $[0,1]$ as well. $\mathcal{N}(\mathcal{S})$ is then equal to the number of non-zero genes of the solution $\mathcal{S}$, divided by a maximal number of emitters able to be added in the area $\mathcal{A}$.

The rest of the AG parameters are classical ones: random population generation, one-point crossover and a random mutation. A design of experiments is in progress in order to define the best settings: population size, probabilities of crossover and mutation, stopping criteria. In order to avoid local optima and to enhance the efficiency of the algorithm, a local search procedure is also applied. This work is in progress and computational experiments are realized to test the efficiency of the proposed resolution approach.

References

1. Yang, S.-H.: Wireless Sensor Networks: Principles, Design and Applications. Springer-Verlag, London (2014)
2. Singh, S.P. and Sharma, S.C.: Range Free Localization Techniques in Wireless Sensor Networks: A Review. Procedia Computer Science, vol. 57, pp. 7-16 (2015)
3. Alshamaa, D., Mourad-Chehade, F. and Honeine, P.: Zoning-based localization in indoor sensor networks using belief functions theory. 2016 IEEE 17th International Workshop on Signal Processing Advances in Wireless Communications (2016)
4. Li, X., Li, D., Wan, J., Vasilakos, A. V., Lai, C.-F. and Wang, S.: A review of industrial wireless networks in the context of Industry 4.0. Wireless Networks, vol. 23(1), pp. 23-41 (2017)
5. Zhao, Y. and Patwari, N.: RFID Localization in Wireless Sensor Networks. Radio Frequency Identification, IntechOpen (2017)
6. Garey, M.R., and Johnson, D.S.: Computers and intractability: A guide to the theory of np-completeness. San Francisco, CA : Freeman, (1979)
7. Meguerdichian, S., Slijepcevic, S., Karayan, V. and Potkonjak, M.: Coverage problems in wireless ad-hoc sensor networks. Proceedings of the IEEE Infocom, vol. 3, pp. 1380-1387 (2001)
8. Yoon, Y., Kim, Y-H.: An Efficient Genetic Algorithm for Maximum Coverage Deployment in Wireless Sensor Networks. IEEE Transactions on Cybernetics, vol. 43(5), pp. 1473 1483 (2013)
9. Gupta, S-K., Kuila, P., Jana, P.: Genetic algorithm approach for k-coverage and m-connected node placement in target based wireless sensor networks. Computers and electrical engineering, vol. 56, pp. 544 556 (2016)
10. Yoon, Y. and Kim, Y.-H.: An Efficient Genetic Algorithm for Maximum Coverage Deployment in Wireless Sensor Networks. IEEE Transactions on Cybernetics, vol. 43(5), pp. 1473-1483 (2013)
11. Khoufi, I., Minet, P., Laouiti, A. and Mahfoudh, S.: Survey of deployment algorithms in wireless sensor networks: coverage and connectivity issues and challenges. International Journal of Autonomous and Adaptive Communications Systems, vol. 10(4) (2017)
12. Le Berre, M., Rebai, M., Hnaien, F. and Snoussi, H.: A Specific Heuristic Dedicated to a Coverage/Tracking Bi-objective Problem for Wireless Sensor Deployment. Wireless Personal Communications, vol. 84(3), pp. 2187-2213 (2015)
13. Smets, P.: Belief functions: the disjunctive rule of combination and the generalized Bayesian theorem. International Journal of approximate reasoning, vol. 9(1), pp. 135 (1993)

Author Index

örnek Mustafa Arslan, 537–547
šilc Jurij, 426–434

Abdelmalik Taleb-Ahmed, 148–154
Abdennacer Bounoua, 148–154
Abderrahim Belmadani, 110–120
Abdul Sattar, 409–418
Adda Chahrazed, 191–203
Aghelinejad Mohsen, 309–311
Ali Lemouari, 188–190
Allali Karam, 127–147
Alves Dos Santos Ramon, 165–174
Amaya-Contreras Iván Mauricio, 54–61
Amodeo Lionel, 325–327
Aoun Oussama, 509–520
Aqil Said, 127–147
Arbaoui Taha, 561–563, 574–576

Baldominos Alejandro, 392–398
Bekhouche Safia, 374–384
Bellala Djamel, 370–373
Benboudjelthia Hafida, 266–272
Bennouar Djamel, 435–444
Benslimane Sidi Mohamed, 210–220
Benttaleb Mourad, 50–52
Benyamina Abou Elhassen, 191–203
Benyettou Mohamed, 2–5
Benyoucef Lyes, 521–530
Benzid Redha, 247–249
Bertoli-Barsotti Lucio, 53
Betka Abir, 31–41
Bouamama Sadok, 477–480
Bouanane Khaoula, 481, 482
Bouazzi Khaoula, 477–480
Boubechal Ikram, 247–249
Bougherara Maamar, 435–444
Boukhobza Abdelkader, 148–154
Boulif Menouar, 361–369
Bouzaachane Khadija, 407, 408
Bouzbita Safae, 232–240
Bouziri Hend, 244–246
Brevilliers Mathieu, 175–177, 204–206
Bue Martin, 283–290

Calenda Tobia, 6–15
Cardoso Pedro, 121–123
Chabchoub Habib, 419–425, 450–473
Chehade Hicham, 561–563, 574–576
Cherif Chahira, 2–5
Chouarfia Abdallah, 19–21
Correa Cely Carlos Rodrigo, 54–61

Couzon Paulin, 531–533
Cruz-Duarte Jorge Mario, 54–61
Cutello Vincenzo, 6–15

Dahane Mohammed, 521–530
De Landtsheer Renaud, 221–231
De Macedo Mourelle Luiza, 165–174, 256–265, 435–444
De V. Cardoso Alexandre, 256–265
Delisle Pierre, 339–350
Denysiuk Roman, 291–300
Deschênes Hugo, 70–72
Di Stefano Antonino, 6–15
Djerou Leila, 250–252
Dkhil Hamdi, 450–473
Duarte Fernando, 291–300
Dupin Nicolas, 314–321

El Afia Abdellatif, 351–360
El Hadaj Salah, 407, 408
El Hami Abdelkhalek, 312, 313
Elafia Abdellatif, 486–495, 509–520
Elhami Norelislam, 474–476
Essaid Mokhtar, 204–206

Fahsi Mahmoud, 210–220
Faramarzi Oghani Sohrab, 283–290
Fatma Zohra Benabdallah, 534–536
Fernades Hortênsio, 121–123
Fiscarelli Antonio Maria, 558–560
Flori Pauline, 325–327
Foderean Daniel, 204–206
Fournier Mathieu, 83–85
Frota Yuri, 63–69

Gagné Caroline, 70–72, 83–85, 339–350
Galvão Dias Teresa, 121–123
Garcia-Perez Arturo, 54–61
Gaspar-Cunha António, 291–300
González-Evstrópova María, 392–398
Gravel Marc, 83–85

Haddou Benderbal Hichem, 521–530
Hadi Anas, 564–573
Hadjadj Mohamed Seddik, 331–338

Hajjem Manel, 244–246
Hamdani Hamid, 312, 313
Hammami Moez, 477–480
Hmina Nabil, 474–476
Hnaien Faicel, 50–52
Hribar Rok, 426–434

Idoumghar Lhassane, 175–177, 204–206
Iguider Adil, 178–187
Ismail Sabrine, 419–425
Itmi Mhamed, 474–476
Ivorra Benjamin, 124–126

Jambi Kamal, 564–573

Kabadurmuş özgür, 548–557
Kalla Hamoudi, 370–373
Kamach Oulaid, 178–187
Kamhuber Felix, 101–109
Kaouter Labed, 253–255
Kemcha Rebiha, 435–444
Kheddouci Hamamache, 331–338
Khelifa Chahinez, 110–120
Khensous Ghania, 19–21
Kouaissah Noureddine, 98–100
Krajecki Michaël, 339–350
Kresta Ales, 95–97

L. Redondo J., 207–209
López Redondo Juana, 124–126
Lahyani Rahma, 419–425
Lalaoui Mohamed, 351–360, 486–495
Lando Tommaso, 53
Leila Djerou, 534–536
Lepagnot Julien, 175–177
Lepagnpt Julien, 204–206
Louaqad Saad, 178–187

M. Ortigosa P., 207–209
Mahammed Nadir, 210–220
Maiza Mohammed, 2–5
Martínez Ortigosa Pilar, 124–126
Martin Emilio, 392–398
Martins Simone, 63–69
Mateus Geraldo, 22–30
Mauceri Stefano, 273–282
Mazière Florian, 339–350
Mcdermott James, 273–282
Mehdi Malika, 266–272
Mellouli Khaled, 244–246
Menezes Gustavo, 22–30
Mesmar Khadija, 127–132
Messabih Belhadri, 19–21

Meurisse Quentin, 221–231
Miyashita Tomoyuki, 496–504
Mjahed Soukaina, 407, 408
Mohamed Ali, 564–573
Mohamed Ben Ali Yamina, 374–384
Mohamed Rahal, 241–243
Mohamed Reghioui, 483–485
Mohamed Ridouan Amattouch, 385–391, 445–449
Morais Ana Catarina, 121–123
Moreno Jorge, 63–69
Mourad-Chehade Farah, 574–576
Moussaoui Manel, 250–252

Nedjah Nadia, 165–174, 256–265, 435–444
Newton M A Hakim, 409–418
Nielsen Frank, 314–321
Nouali Taboudjemat Nadia, 266–272
Noumbissi Tchoupo Moïse Aimé, 325–327

Oesterle Jonathan, 561–563
Ortobelli Sergio, 98–100
Ouazene Yassine, 309–311, 531–533
Ouldkradda Ali, 210–220
Ourlis Lazhar, 370–373
Ousmer Sabrine, 266–272

Pérez-Sánchez H., 207–209
Papa Gregor, 426–434
Parque Victor, 496–504
Parsopoulos Konstantinos, 86–94
Pavão Leandro, 158–164, 301–308
Pavone Mario F., 6–15
Petelin Gašper, 426–434
Pinto Rene, 291–300
Polash M M A, 409–418
Ponsard Christophe, 221–231
Puertas-Martín S., 207–209

Radi Bouchaïb, 312, 313
Raghay Said, 407, 408
Rahimi Mohammad, 155–157
Rakhshani Hojjat, 175–177
Ramos ángel, 124–126
Ravagnani Mauro, 158–164, 301–308
Riahi Vahid, 409–418
Riane Fouad, 505–508
Rodoplu Melek, 328–330
Ruiz Ferrández Miriam, 124–126

Saadi Hocine, 266–272
Sabiri Asmaa, 505–508
Sadeghi Rezvan, 561–563

Saez Yago, 392–398
Santos Lucas, 22–30
Sarubbi João, 22–30
Satoh Ichiro, 73–82
Sbaa Salim, 31–41
Schieder Peter, 101–109
Seghir Rachid, 247–249
Selmi Aymen Takie Eddine, 250–252
Sfaksi Sara, 250–252
Sihn Wilfried, 101–109
Silva Hugo, 291–300
Sioud Aymen, 83–85
Sobottka Thomas, 101–109
Steinbuch Rolf, 16–18
Sweeney James, 273–282

Talbi El-Ghazali, 6–15, 155–157, 244–246, 283–290, 314–321
Taleb-Ahmed Abdelmalik, 2–5
Tamura Kenichi, 322–324
Tatsis Vasileios, 86–94
Terki Nadjiba, 31–41
Tetouani Samir, 42–49
Thivet Eric, 325–327
Tichy Tomas, 62
Torres Bispo Dos Santos Leonardo, 165–174
Torres Leonor, 121–123
Toumi Abida, 31–41
Touzout Fayçal A., 521–530

Ulrich Maximilian, 101–109

Varlet Eric, 283–290
Vitale Alessandro, 6–15
Vukašinović Vida, 426–434

Yalaoui Alice, 309–311, 325–327, 483–485
Yalaoui Farouk, 50–52, 325–327, 531–533, 561–563, 574–576
Yamina Mohamed Ben Ali, 399–406
Yassine Adnan, 450–473

Zemzami Maria, 474–476
Zouadi Tarik, 483–485

Sponsors

informatiques mathématiques
Inria
cnrs
Université de Lille
EMI
gdr macs
CRIStAL
Centre de Recherche en Informatique, Signal et Automatique de Lille
SYNERGY
Horizon 2020
GA No 692286
GdR RO
CNRS - GdR 3002
LIRIMA