International Journal of Information Retrieval Research
Volume 12 • Issue 1

Solving the Cubic Cell Formation Problem Using Simulated Annealing

Hamida Bouaziz, Mecatronique Laboratory, University of Jijel, Algeria*
Ali Lemouari, LMAM Laboratory, University of Jijel, Algeria

ABSTRACT

The cubic cell formation problem (CCFP) in cellular manufacturing systems consists of decomposing a production system into a set of manufacturing cells and assigning workers to cells besides parts and machines. The major objective is to obtain manageable cells. Manageable cells mean cells with a minimum value of inter-cell moves of parts and workers and a minimum value of heterogeneity within cells. In this paper, a solution methodology based on a modified simulated annealing heuristic with a proposed neighbourhood search procedure is proposed. The methodology allows building multiple configurations by giving to the decision-maker the ability to control some parameters. Experimental results show that the proposed algorithm gives a promising performance for all problem instances found in the literature.

KEYWORDS

Cubic Cell Formation Problem, Manufacturing Cells, Neighbourhood Search Procedure, Production System, Simulated Annealing

INTRODUCTION

The Group Technology (GT) is a manufacturing concept that seeks to identify and group similar parts to take advantage of their similarities in system manufacturing and design. It has been practiced for many years around the world as part of good engineering practice. A Cellular Manufacturing System (CMS) is an application of the group technology; it is used to design the layouts of production systems. The problem of cell formation (CFP) in cellular manufacturing systems (CMSs) is an important problem in the operational research literature (Joines, King, & Culbreth, 1996), (Nourie, Tang, Tuah, Ariffin, & Samin, 2013). It consists on decomposing an entire production system into a set of manufacturing cells, and assigning the machines and allocating the parts, to be produced, to these production cells. During this decomposition, some constraints and objectives must be considered to produce most manageable and independent cells.

Considering only the minimization of inter-cell moves without constraints on the number of the cells produces a design with a single cell for the production system. By this way, the advantages of cellular manufacturing system will be lost. It is well known that when designing a cellular manufacturing system, the objective is to set up manageable cells by assigning to them parts and

DOI: 10.4018/IJIRR.290827 *Corresponding Author

machines at a manageable level. To obtain manageable cells, two objectives are considered in this study: the minimisation of inter-cell moves of parts and workers, and the maximisation of machine use within cells by minimising heterogeneity of cells. These two objectives are considered as conflicting objectives because when trying to reduce the inter-cell moves, many machines are regrouped in a small number of cells. This contributes into incrementing the heterogeneity within cells. On the other hand, the maximum inter-cellular moves will be created at a minimum value of heterogeneity because the minimum value of heterogeneity is obtained when each cell is dedicated for each type of machines, and thus a part to be processed needs to visit a new cell at each processing operation.

Real situations are, generally, constrained by some limitations which are often difficult to quantify. These limitations make difficult the implementation to reach the best solution. Having alternative configurations and giving the decision-maker the ability to control the best trade-off between the two conflicting objectives is a great advantage in such conditions (Shiyas & Pillai, 2014). In this study, to let the decision-maker produces such configurations a weight parameter is used to regulate the importance of heterogeneity.

This paper focuses on a variant of the cell formation problem, known as the cubic cell formation problem (CCFP). In CCFP, three links must be considered: the relation between the parts and the machines, the relation between the machines and the workers, and the relation between the workers and the parts (an example is shown in Table 1). The assignment of workers to cells is considered besides the affectation of parts and machines. Multiple configurations and automatic generation of the number of cells are also considered. In the literature, many techniques and algorithms have been proposed to solve the CFPs, including heuristics, meta-heuristics, exact methods, etc. (Nourie, Tang, Tuah, Ariffin, & Samin, 2013). Using exact methods to solve CFPs allows obtaining the best-existed solutions, but due to the NP-hard nature of these combinatorial problems, when the dimensions of the problem increase, these exact methods become extremely costly in term of time and memory consumption. For that reasons, meta- heuristic techniques are considered more convenient to solve NP-hard problems and to produce good solutions in reasonable time (Bouaziz & Lemouari, 2020). In this study, a Simulated Annealing (SA) algorithm, with a proposed neighbourhood search procedure, is used to solve the CCFP. SA was applied to various problems (Jiang, Ji, Lu, Y., & Jia, 2019), (Leite, Melício, & Rosa, 2019), and (Tasoglubc & Yildiz, 2019) including the basic version of CFP (Wu, Chung, & Chang, 2009) and has given very good results.

The remainder of this paper is structured as follows: In section II, we present related work. In Section III, the mathematical model is presented. In Section IV, the simulated annealing algorithm and the neighbourhood structure are detailed. In Section V, the computational results are shown, and a comparison with existed methods is performed. Finally, in Section VI, the conclusion and future perspectives are given.

RELATED WORK

In the literature, a vast variety of CFPs have been described. Many studies focus on the two dimensional manufacturing cell formation problems. Although the importance of the human dimension, the studies

Table 1. Example of cubic cell formation problem

Parts-Machines		Machines-Workers		Workers-Parts	
Parts	Machines	Machines	Workers	Workers	Parts
	1 2 3 4		1 2 3 4		1 2 3 4
1	0 1 1 1	1	1 1 0 1	1	1 1 0 1
2	1 0 1 0	2	1 1 1 1	2	1 1 1 0
3	1 1 1 1	3	1 1 1 1	3	1 1 1 1
4	1 0 1 0	4	0 1 0 0	4	1 1 1 1

consider only the part and the machine dimensions. However, the worker dimension is neglected. When solving this basic kind of the problem, two operations must be performed which are the grouping of similar parts into part families and machines into machine cells. In (Wu, Chung, & Chang, 2009), a hybrid method of the simulated annealing algorithm with the mutation operator of genetic algorithm is used to solve the two dimensional cell formation problem. In (Shiyas & Pillai, 2014), A two dimensional cell formation problem is solved using Genetic Algorithm (GA) meta-heuristic. The authors deal with multiple configurations without considering the worker dimension. Also, their notion of heterogeneity is a bit different because it does not take into consideration the cells of the parts but the cells must be visited by the parts. In (Danilovic & Ilic, 2019), the authors developed a new hybrid algorithm CFOPT (Cell Formation OPTimization) to solve the basic cell formation problem. Their mechanism consists in using the specificity of the input instances to narrow down the feasible set of solution in order to increase the efficiency of the optimization process. Mahmoodian, Jabbarzadah, Rezazadeh, & Barzinpour (2019) presented a new PSO-based algorithm (Particle Swarm Optimization based algorithm) to solve the basic cell formation problem. The algorithm integrates the self-organization map neural networks to PSO algorithm. In (Karoum & Elbenani, 2019), the authors combined a local search mechanism with cuckoo search algorithm in order to intensify the search and improve the grouping efficacy of the solutions. All the above mentioned studies do not consider the workers that handle the machines.

Limited studies of CFP considering the human dimension can be found. The cubic cell formation problem, that includes the worker as third dimension, has been first introduced by Min and shin (1993). In (Mahdavi, Aalaei, Paydar, & Solimanpur, 2012), the authors used the branch-and-bound (B&B) method under Lingo software to solve the model of cubic cell formation problem. In their model, the two considered objectives are the minimisation of voids and exceptional elements. In (Nikoofarid & Aalaei, 2012), the authors presented a new mathematical model for a cell formation problem in production planning in a dynamic virtual cellular manufacturing system. The proposed model includes the worker dimension, and it considers as objectives the minimization of the holding and backorder costs and the management of machines and workers over a certain planning horizon. In (Aalaei & Shavazipour, 2013), the authors defined an integer mathematical programming model to design the cellular manufacturing systems under data envelopment analysis. They attempted to minimize backorder costs and inter-cellular movement cost caused by exceptional elements. In (Bootaki, Mahdavi, & Paydar, 2014), the author used a hybrid GA-Augmented ε-constraint method to solve the generalized cell formation problem. In (Bootaki, Mahdavi, & Paydar, 2015), the authors developed a new multi-objective mathematical model to design dynamic cubic binary cell formation problem. In their model, the authors consider the machine and the worker utilisation concept. To solve their model, the authors developed a new goal programming method called "percentage multi-choice goal programming" (PMCGP). In (Sahin & Alpay, 2016), a genetic algorithm is developed to solve CCFP. GA is characterized by a large number of parameters that need a huge effort to tune them properly. Thus, to set the appropriate level of these parameters, Taguchi as a statistical method is used by the authors. In (Feng, Da, Xi, Pan, & Xia, 2017), A hybrid approach combining Particle Swarm Optimisation (PSO) and linear programming (LP) is used to solve the CCFP. In (Bouaziz & Lemouari, 2020), a discrete flower pollination algorithm is developed to solve the cubic cell formation problem. The solved version of CCFP is not the binary one. The algorithm takes the number of cells as entry and it cannot decide on the number of cells.

In this study, the focus is made on solving the cubic version of cell formation problem. As in (Nikoofarid & Aalaei, 2012) and (Shiyas & Pillai, 2014), the developed algorithm is able to generate automatically the number of cells resulting in the best configuration of the system. Nikoofarid & Aalaei (2012) developped a multiobjective algorithm in order to generate a set of non dominated solutions which are considered as a set of possible configurations. However, in this study in order to produce a set of possible configurations, we have implemented the same technic used by Shiyas & Pillai (2014), which consists in giving the decision maker the ability to control a parameter of the

objective function in order to implement his preferences. Table 2 shows a comparison of this study with related work. The comparison considers three criteria which are: (i) the worker dimension, (ii) the multiple configurations and (iii) the automatic generation of the number of cells.

THE MATHEMATICAL MODEL

In the basic CFP, only two dimensions are considered which are the part and the machine dimensions. To solve the basic CFP, the relationships between machines and parts must be given as inputs. These relationships are represented as a binary machine-part incidence matrix A= [a_{ij}], where the entry $aij_{,\text{ of}}$ the matrix A, takes the value 1 if machine i processes part j, otherwise, it takes the value 0 (Bouaziz & Lemouari, 2020). When dealing with CCFP a third dimension is considered which is the worker dimension. As input, for every problem instance three binary matrices must be provided: parts-machines $\left[a_{pm}\right]$, machines-workers $\left[b_{mw}\right]$, and workers-parts $\left[c_{wp}\right]$. The p*arts-machines* matrix establishes the relation between the parts and the machines. It specifies the set of machines needed by each part to be processed. The m*achines-workers* matrix specifies the workers that can handle each machine. The w*orkers-parts* matrix establishes the relation between the workers and the parts. It indicates, for each worker, the set of parts that he may contribute in their processing.

Solving the problem consists in taking four decisions: The first decision concerns the specification of the cell of each part $\left(X_{pk}\right)$. The second decision indicates the cell of each machine $\left(Y_{mk}\right)$. The third decision depicts a cell for each worker $\left(Z_{wk}\right)$. Finally, the last decision indicates the worker that must process each part, on each machine needed by this part, and within which cell $\left(d_{pmwk}\right)$. In this

Table 2. Comparison with related work

The study	Worker Dimension	Multiple configurations	Automatic generation of the number of cells	Resolution method
(Bouaziz & Lemouari, 2020)	×			DFPA
(Karoum & Elbenani, 2019)				Cuckoo
(Danilovic & Ilic, 2019)				CFPOT
(Mahmoodian, Jabbarzadah, Rezazadeh, & Barzinpour, 2019)				PSO
(Feng, Da, Xi, Pan, & Xia, 2017)	×			PSO& LP
(Sahin & Alpay, 2016)	×			GA
(Bootaki, Mahdavi, & Paydar, 2015)	×			PMCGP
(Bootaki, Mahdavi, & Paydar, 2014)	×	×		GA-AUG
(Shiyas & Pillai, 2014)		×	×	GA
(Aalaei & Shavazipour, 2013)	×			-
(Mahdavi, Aalaei, Paydar, & Solimanpur, 2012)	×			B&B
(Nikoofarid & Aalaei, 2012)	×			-
(Wu, Chung, & Chang, 2009)			×	HSAM
Our Study	×	×	×	SA

study, two objectives must be considered during the resolution. The first one is the number of exceptional elements (*EE*), it refer to the need to move parts from their cells to other cells. The second objective is the number of voids (*H*). It measures the number of machines and workers grouped with parts that these last do not need for processing.

The model used in this study is also considered in (Mahdavi, Aalaei, Paydar, & Solimanpur, 2012) and in (Sahin & Alpay, 2016) to model the CCFP. To deal with multiple configurations, an adaptation of the objective function is established by multiplying the number of voids (heterogeneity) by a weight parameter. This parameter may be controlled by the designer to produce the suitable configurations to real situations.

Notations

- C: the total number of cells.
- M: the total number of machines.
- P: the total number of parts.
- W: the total number of workers.
- k: the index of cells, k= 1,2, ...,C.
- p: the index of parts, p=1,2, ...,P.
- m: the index of machines, m=1,2, ...,M.
- w: the index of workers.
- U, L: the maximum and the minimum cell size in term of machines.
- LP, LW: the minimum cell size in term of parts and workers.
- a_{pm} : a binary parameter indicating whether part p is processed on machine m.
- b_{mw} : a binary parameter indicating whether worker w may work on machine m.
- c_{wp} : a binary parameter indicating whether worker w may participate in producing p.
- EE: the total number of exceptional elements (inter-cellular moves) in the solution.
- H: the total number of voids in the solution (heterogeneity).
- γ: the parameter used to regulate the importance of heterogeneity of the formed design.

Decision Variables

Four binary decision variables are used in the mathematical model:

$$X_{pk} = \begin{cases} 1, & if\ part\ p\ is\ allocated\ to\ cell\ k \\ 0, & otherwise \end{cases}$$

$$Y_{mk} = \begin{cases} 1, & if\ machine\ m\ is\ assigned\ to\ cell\ k \\ 0, & otherwise \end{cases}$$

$$Z_{wk} = \begin{cases} 1, & if\ worker\ w\ is\ assigned\ to\ cell\ k \\ 0, & otherwise \end{cases}$$

$$d_{pmwk} = \begin{cases} 1, & if\ part\ p\ is\ processed\ on\ machine\ m\ by\ worker\ w\ within\ cell\ k \\ 0, & otherwise \end{cases}$$

The Model

$$\min\ f = EE + \gamma H \tag{1}$$

where:

$$EE = \sum_{p=1}^{P}\sum_{k=1}^{C}\sum_{m=1}^{M}\sum_{w=1}^{W} d_{pmwk} Y_{mk} \left(2 - X_{pk} - Z_{wk}\right) \tag{2}$$

$$H = \sum_{k=1}^{C}\sum_{p=1}^{P}\sum_{m=1}^{M}\sum_{w=1}^{W} (1 - d_{pmwk}) X_{pk} Y_{mk} Z_{wk} \tag{3}$$

subject to:

$$\sum_{k=1}^{C} X_{pk} = 1,\ \forall p \tag{4}$$

$$\sum_{k=1}^{C} Y_{mk} = 1,\ \forall m \tag{5}$$

$$\sum_{k=1}^{C} Z_{wk} = 1,\ \forall w \tag{6}$$

$$dpmwk \le a_{pm} b_{mw} c_{wp} Y_{mk},\ \forall p, m, w, c \tag{7}$$

$$L \le \sum_{m=1}^{M} Y_{mk} \le U,\ \forall k \tag{8}$$

$$\sum_{p=1}^{P} X_{pk} \ge LP,\ \forall k \tag{9}$$

$$\sum_{w=1}^{W} Z_{wk} \ge LW,\ \forall k \tag{10}$$

$$\sum_{k=1}^{C}\sum_{w=1}^{W} d_{pmwk} = a_{pm},\ \forall p,m \tag{11}$$

$$X_{pk}, Y_{mk}, Z_{wk}, d_{pmwk} \in \{0,1\},\ \forall p,m,w,k \tag{12}$$

The objective function given in Equation (1) minimises the inter-cell moves and weighted heterogeneity. The number of exceptional elements is computed using equation (2), it includes the movement of parts and workers between cells. However, equation (3) calculates the heterogeneity. Equation (4), (5), (6) ensures, respectively, that each part, each machine, and each worker is assigned to exactly one cell. Equation (7) controls the availability of machine m in cell k. Equation (8) gives restrictions on the size of the cells in term of the number of machines. Equation (9), (10) respectively ensures that each cell must contain at least LP parts and LW workers. Finally, equation (11) specifies that exactly one worker is assigned to produce a given part p on a given machine m if p needs to be processed on m. The last equation specifies that decision variables are all binary variables.

SOLUTION METHODOLOGY

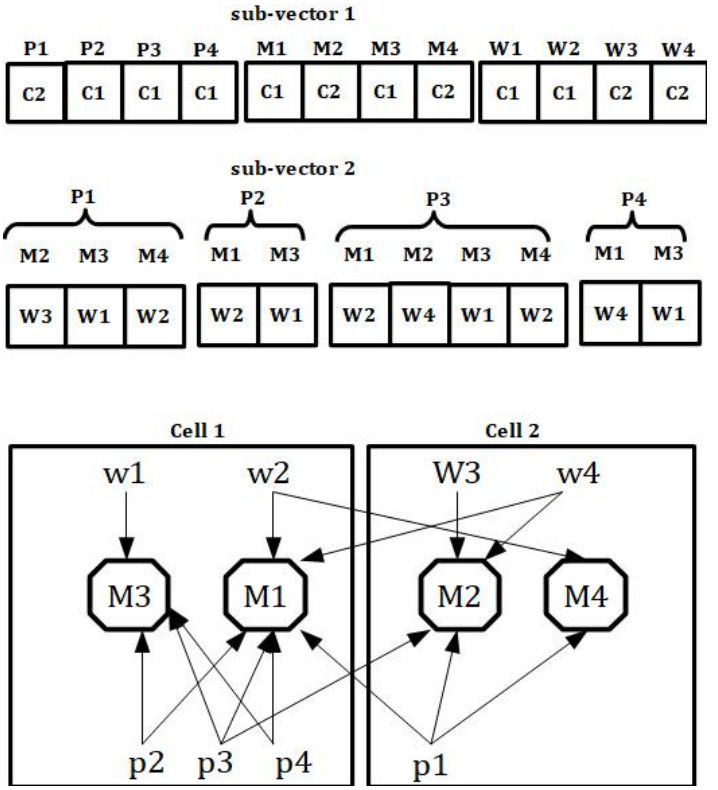
In this study, SA (Simulated Annealing) algorithm is used to solve the model of cubic cell formation problem. The focus was made on the mechanism used to move from a solution to another one in the search space. SA is a probabilistic optimization technique for approximating the global optimum of a function. It has been applied to many optimization problems including the basic cell formation problem (Wu, Chung, & Chang, 2009) with a high success. Compared to other stochastic search methods, SA stands out by its fast convergence to provide optimal or near optimal solutions. When solving the cubic cell formation problem using SA, the generated solution must respect the decider choices in term of how match the heterogeneity is considered. The solution specifies: (1) the cell of each machine, (2) the cell of each part, (3) the cell of each worker, and (4) the responsible worker of executing each operation of each part, on which machine and within which cell (the cell must contain the corresponding machine because the machines could not be shift from their cells).

Solution Representation

The solution is represented using a vector. This vector includes two sub-vectors: the first one has a size equal to $P+M+W$. It implements the decisions about the cells to which each part, machine and worker is assigned. The first piece of this sub-vector has a size equal to P; it represents the cell to which each part is affected. The second piece specifies the cell to which each machine is assigned, thus it has a size equal to M. However, the last piece has a length equal to W, and it indicates the cell to which each worker is assigned.

The second sub-vector implements the decision about the assignment of workers to the part operations. It specifies which worker must perform each operation of each part and on which machine. In this sub-vector, there is no need to specify the cell within the processing will be performed because this information is already available in the first sub-vector. The processing is executed within the cell where the machine is assigned, because in CFP machines cannot be shift from a cell to another. Thus, the size of this sub-vector is equal to the sum of the number of machines needed by every part. A feasible solution for the problem shown in Table 1 is given in Figure 1. In this solution, part 2, part 3, part4, machine 1, machine 3, worker 1 and worker 2 are assigned to cell 1. However, the remaining parts, machines and workers are assigned to cell 2. From sub-vector 2, it is clear that part

Figure 1. An example of a solution for a CCFP (Bouaziz & Lemouari, 2019)



1 on machine 2 will be processed by worker 3 within cell 2. It is in cell 2 where the processing is performed because machine 2 is assigned to cell 2.

Quality of Solutions

For a better exploration of the search space, the objective function (Equation 1) is modified by applying the factor $\beta.IC$, equation 1 becomes:

$$\min \ f = EE + \gamma \cdot H + \beta \cdot IC \tag{13}$$

where:

- The factor $\beta.IC$ of function f allows penalizing the infeasible solutions.
- IC is the number of poor cells. A cell is qualified as a poor cell if it does not respect the constraints controlled its size in terms of machines, parts or workers.
- β is a parameter used to adjust the impact of the number of poor cells on the quality of the solution. To tolerate the violation of constraints that control the size of cells, this parameter should take a small value. Otherwise, a large value should be assigned to this parameter.

The value of parameter γ has an impact on the generated configuration (solution). The variation of the value of this parameter allows obtaining several configurations. To produce a configuration that privileges the number of exceptional elements over the value of heterogeneity, a small value must

be assigned to γ. Otherwise, the value of γ should be increased. The value of γ has a direct impact on the number of cell resulting in the best objective values which is generated automatically in this study. The number of cells may take a value which is comprised between éM/Uù and ëM/Lû. By experiment, it is found that by increasing the number of cells, the objective value becomes better until a given threshold *t*. After that, when *C* varies in the interval $\left[t,\left\lfloor M\ /\ L\right\rfloor\right]$, the objective function *f* takes always the same value or even worse. Thus, our algorithm is stopped when reaches *t*, or when attempts a maximum number of iterations without any improvement in the solution.

Candidate Worker Matrix

The candidate Worker Matrix (CWM) is computed using the three input matrices: parts-machines $\left[a_{pm}\right]$, machines-workers $\left[b_{mw}\right]$, and workers-parts $\left[c_{wp}\right]$. It specifies, for each part and each machine, the feasible set of workers who will be able to produce this part on the specified machine. This matrix makes easier discovering the feasible search space, when dealing with the second sub-vector of the solution. The candidate worker matrix of the problem presented in Table 1 is shown in Table 3.

For example, when part 1 needs to be processed on machine 4, there is only one worker, which is worker 2 that can execute this operation. However, the same part may be processed by any worker on machine 2 and machine 3.

Neighbourhood Structure

For a best exploration of the search space, a Candidate Move List (CML) is created to contain the set of possible moves that may be applied on the current solution to produce a neighbour. Each move is defined as a tuple *M*(*position*, old_v, new_v), where *position* points the box on the solution vector that must change its value from old_v to new_v if this move is selected to create the neighbourhood solution.

In order to discover the search space, four types of moves are defined:

- Move Part M1(*p*, old_v, new_v) where $0 \le p < P$, and old_v, $new_v \in C$, consists in changing randomly the cell of part p.
- Move Machine M2(*p*, old_v, new_v) where $P \le p < P + M$, and old_v, $new_v \in C$, consists in shifting machine $p - P$ to a random cell.
- Move Worker M3(*p*, old_v, new_v) where $P + M \le p < P + M + W$, and $old_v, new_v \in C$, consists in moving worker $p - \left(P + M\right)$ to a random cell.

Table 3. Candidate worker matrix for the presented example

Part	Machine	Worker
P1	M2 M3 M4	1 2 3 4 1 2 3 4 2 0 0 0
P2	M1 M3	1 2 4 0 1 2 3 4
P3	M1 M2 M3 M4	2 4 0 0 2 3 4 0 2 3 4 0 2 0 0 0
P4	M1 M3	1 4 0 0 1 3 4 0

- Assign Worker M4(p, old_v, new_v) where $P + M + W \leq p$, and $old_v, new_v \in W$, consists in modifying the worker that must perform the operation of the concerned part by the concerned machine. To avoid generating infeasible solution by selecting a worker that cannot perform the concerned operation, the candidate worker Matrix (CWM) represents an important reference.

At any iteration of the SA procedure, only one move is applied. This move is selected according to the strategy detailed in Algorithm 1.

Algorithm 1. Create neighbourhood solution

```
1:  for each move M ∈ CML do
2:     Apply m on the current solution S_c
3:     Calculate f(S_c) and add it to the list f_CML
4:     Apply the inverse move of M to restore the current solution Sc
5:  end for
6:  Calculate max and mean of the list f_CML
7:  Calculate the threshold tf = max − λ*(max − mean), where λ ∈ [0 − 1]
8:  for each move M in CML do
9:     if f_CML(M) ≥ tf then
10:       Remove m from CML
11:    end if
12: end for
13: Select a random move M in CML to be applied on the current
    solution to build the neighbourhood solution.
```

By restricting the size of the candidate movement list (Lines [8-12]), the worst neighbours are discarded from the list of all potential neighbours. The main motivation behind this step is improving the search quality, and reaching the optimum more quickly.

SA ALGORITHM

The following notations are used:

- T_0 : the initial temperature
- A : the cooling rate
- S_0 : the initial solution
- S_c : the current solution
- S_n : the neighbourhood solution
- S^* : the incumbent solution of current cell size
- S^{**} : the best solution
- C^{**} : the best number resulting in best solution
- *NI*: the counter that specifies the allowed number of iterations without improving the incumbent solution. Its initial value is set at $\max_{ni}$.
- *ST*: the number of times a neighbourhood solution is generated in a specific temperature. Its maximum value is equal to $\max_{st}$.

The algorithm starts by a step of initialization. After that, a routine is repeated until the number of cell reaches its maximum or when the aforementioned threshold of the number of cells is reached. At each iteration, a classical simulated annealing routine is triggered in order to obtain a configuration with the concerned number of cells. At the end of the iteration, if the incumbent solution of current cell size S^* is better than the best found solution S^{**} then it takes its place. The detail of our algorithm is provided at the level of Algorithm 2:

Algorithm 2. SA Algorithm

```
1:  C ← ⌊M / U⌋,  C** ← C,  f(S**) ← ∞
2:  while C ≤ ⌊M / L⌋ do
3:      Initialise T, α, max_ni, max_st
4:      Create random solution s0. Let S_c ← S_0,  S* ← S_0
5:      while NI ≠ 0 do
6:          S_n ← Create_neighbourhood_solution(S_c)
7:          if f(S_n) < f(S_c) then
8:              S_c ← S_n
9:              if f(S_n) < f(S*) then
10:                 S* ← S_n
11:                 NI ← max_NI
12:             else
13:                 NI ← NI-1
14:             end if
15:         else
16:             NI ← NI-1
17:             r ← random(0,1)
18:             ΔF ← f(S_n) - f(S_c)
19:             if r < e^(-(ΔF/T)) then
20:                 S_c ← S_n
21:             end if
22:         end if
23:         ST ← ST+1
24:         if ST = max_st then
25:             T ← α * T
26:             ST ← 0
27:         end if
28:     end while
29:     if f(S*)<f(S**) then
30:             C** ← C;  S** ← S*
31:             C ← C+1
32:     end if
33: end while
```

In line 1, the initial number of cells is set at M / U. M / U represents the lowest number of cells that may be constructed. At the end of every run of SA procedure (line 31), the number of cells

is increased by 1 as long as the solution is improved and the number of cells does not exceed M / L. In line 3, SA procedure is restarted by reinitializing the SA parameters (T, α …). The NI variable takes initially a value which is equal to the tolerated number of times that SA procedure iterates without improvement. NI is decreased by 1 each time SA fails to enhance the incumbent solution (line 13 and 16). When the variable NI reaches the value 0, the SA procedure triggered for the current number of cells is stopped. As long as NI is greater than 0, a new neighbourhood solution is created by applying one move from the four previously defined moves (line 6). If the neighbourhood solution is better than the current solution, the neighbourhood solution replaces the current solution. Otherwise, the neighbourhood solution is accepted by a probability which is equal to $e^{-(\Delta F/T)}$. If the neighbourhood solution is better than the current solution, another comparison is done between this neighbourhood and the incumbent solution of current cell size (line 9). If the neighbourhood solution is better than the incumbent solution, it takes its place. At each iteration of the algorithm, if the incumbent solution improves the best solution, than the best solution is replaced by the incumbent solution, the number of cells is increased, and another SA procedure is triggered for the new number of cells.

COMPUTATIONAL RESULTS AND DISCUSSION

The simulated annealing algorithm was coded in java and run on a PC Intel(R) Core(TM) i3-5010U CPU running at 2.10 GHz with 4 GB of RAM. Using the trial and error method, and after intensive testing, the parameters are set as follows: T_0 =2000, α=0.998, $\max_{st}$ =10, NI=20000, λ=0.6.

Multiple Configurations

To show the utility of having multiple configurations as solutions for the same problem, the algorithm is run on the problem presented in Table 1 for different values of γ in the interval [0, 10]. Table 4 shows that for a zero value of γ, a configuration with a single cell is obtained. The solution in this case favours the intercellular movement objective over the heterogeneity objective. At a γ value equal to 0.7 or greater, a configuration with 3 cells results in the best objective values. The obtained solution promotes the heterogeneity objective over the exceptional elements objective. Between these two values of γ another configuration is obtained. In this case, the configuration exhibits almost the same value for the two objectives.

These configurations may differ in the number of cells, the value of exceptional elements, or the value of heterogeneity. The contributions of each element to the objective function, within each configuration presented in Table 4, are shown in Table 5. Each box in the table contains a tuple where the first element represents the value of exceptional elements. However, the second element of the tuple represents the value of heterogeneity produced by the concerned part, machine and worker. For example, at the level of the second configuration, the box at the intersection of part 1, machine 3 and worker 4 contains the tuple (1,0). This means that there is one exceptional element. This later is caused by part 1 which is affected to cell 2, however machine 3 and worker 4 are assigned to cell 1. Thus, part 1 needs to be shift to cell 1 in order to be processed.

Table 4. Different configurations obtained for different values of γ (Bouaziz & Lemouari, 2019)

γ	C	H	EE	F	Solution
0	1	53	0	0	1111111111113243222241
0.1	2	4	3	3.4	21111212222134244434244
0.7	3	1	5	5.7	13233231112334244434244

Table 5. The detail of each configuration

First configuration																
	Part1				Part2				Part3				Part4			
	M_1	M_2	M_3	M_4	M_1	M_2	M_3	M_4	M_1	M_2	M_3	M_4	M_1	M_2	M_3	M_4
W_1	0 , 1	0 , 0	0 , 1	0 , 1	0 , 1	0 , 1	0 , 1	0 , 1	0 , 1	0 , 1	0 , 1	0 , 1	0 , 1	0 , 1	0 , 0	0,1
W_2	0 , 1	0 , 1	0 , 1	0 , 0	0 , 1	0 , 1	0 , 1	0 , 1	0 , 0	0 , 0	0 , 0	0 , 0	0 , 1	0 , 1	0 , 1	0,1
W_3	0 , 1	0 , 1	0 , 0	0 , 1	0 , 1	0 , 1	0 , 0	0 , 1	0 , 1	0 , 1	0 , 1	0 , 1	0 , 1	0 , 1	0 , 1	0,1
W_4	0 , 1	0 , 1	0 , 1	0 , 1	0 , 0	0 , 1	0 , 1	0 , 1	0 , 1	0 , 1	0 , 1	0 , 1	0 , 0	0 , 1	0 , 1	0,1
Second configuration																
	Part1				Part2				Part3				Part4			
	M_1	M_2	M_3	M_4	M_1	M_2	M_3	M_4	M_1	M_2	M_3	M_4	M_1	M_2	M_3	M_4
W_1	0 , 0	0 , 1	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
W_2	0 , 0	0 , 1	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
W_3	0 , 0	0 , 0	0 , 0	0 , 1	0 , 0	0 , 0	0 , 0	0 , 0	0 , 0	1 , 0	0 , 0	0 , 0	0 , 0	0 , 0	0 , 0	0,0
W_4	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 , 0	0,0
Third configuration																
	Part1				Part2				Part3				Part4			
	M_1	M_2	M_3	M_4	M_1	M_2	M_3	M_4	M_1	M_2	M_3	M_4	M_1	M_2	M_3	M_4
W_1	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
W_2	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
W_3	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 , 0	0 , 0	0,0
W_4	0 , 0	0 , 0	1 , 0	0 , 0	0 , 0	0 , 0	0 , 0	0 , 0	1 , 0	0 , 0	1 , 0	0 , 0	0 , 0	0 , 0	0 , 0	0,0

The selection of a particular configuration is controlled by the practical consideration and the choices of the decision-maker for the number of cells, and the values of the two objectives (exceptional elements and heterogeneity) included in the objective function. Thus, the decision-maker can select the appropriate configuration for the real situation.

Generating Automatically the Number of Cells

Giving the designer the possibility of generating automatically the number of cells, resulting in the best objective values for the problems, represents a great advantage when the number of cells does not represent a constraint that prevents the realization of a given design.

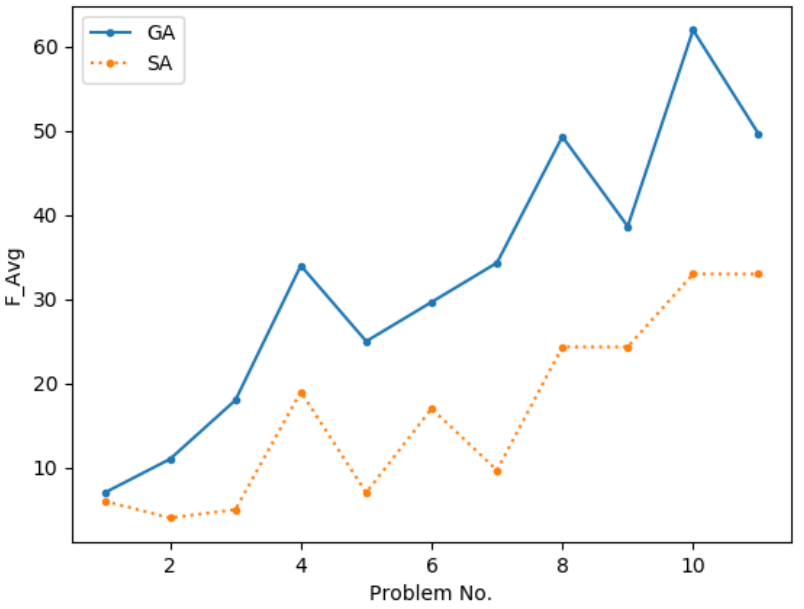
The proposed SA algorithm is tested on the 11 instances available in (Sahin & Alpay, 2016). By considering the number of cells as a decision variable, SA calculates the number of cells resulting in the best objective values for the problems. Table 6 shows the gap between the objective values obtained by SA when the decision on the number of cells is taken during the resolution, and the objective values obtained by GA (Sahin & Alpay, 2016) where the number of cells is fixed at a given integer before running the algorithm. Table 6 shows that the number of cells has a great impact on the objective value of the obtained solution.

The average of the objective value obtained when running the algorithm tree times on each test instance is scattered in Figure 2. For each test instance, the obtained solution when applying the simulated annealing algorithm is better than the one obtained by the genetic algorithm but with different number of cells.

Table 6. Determine the best number of Cells (Bouaziz & Lemouari, 2019)

PBM (P×M×W)	GA			SA			Gap(%)
	C	F_Best	F_Avg	C	F_Best	F_Avg	
p1 (4,4,4)	2	7	7	3	6	6	14.28
p2 (5,4,5)	2	11	11	3	4	4	63.64
p3 (6,5,5)	2	18	18	4	5	5	72.22
p4 (10,7,4)	2	34	34	4	19	19	44.12
p5 (10,7,6)	3	25	25	6	7	7	72.00
p6 (12,8,6)	3	29	29.66	6	17	17	42.68
p7 (12,8,7)	3	33	34.33	7	9	9.67	71.83
p8 (15,10,6)	3	49	49.33	6	24	24.33	50.70
p9 (15,10,6)	4	37	38.66	6	24	24.33	37.07
p10 (20,10,6)	3	60	62	6	33	33	46.77
p11(20,10,6)	4	47	49.66	6	33	33	33.55

Figure 2. The results of considering the number of cells as decision variable



Fixed Number of Cells

To perform a fair comparison between the results of SA, the results reported by Yelliz, and those obtained by GAMS software (Sahin & Alpay, 2016), the value of γ is set to be equal to 1. Also, the number of cells is fixed according to the values mentioned in the study of Sahin & Alpay (2016). Thus, the objective value of the obtained solution and the execution time are reported in Table 7.

In Table 7, the term GAP measures the deviation between the results of SA implemented in this study and the results of GAMS software and GA presented in (Sahin & Alpay, 2016). The gap between SA and GAMS is computed according to Equation 14:

$$Gap_{GAMS-SA} = \frac{F_{GAMS} - F_{Avg(SA)}}{F_{GAMS}} * 100 \quad (14)$$

However, the gap between SA and GA is calculated according to Equation 15.

Considering the objective function values, the performance of the implemented SA on all of the test problems is equal or better than those of the GA algorithm and the GAMS software. The out-performance of SA appears clearly when dealing with large sized test problems where the deviation takes always a positive value. Especially for test instance #9, the gap between simulated annealing algorithm and LINGO software reached 10.26%. This value may be justified by the large size of the search space, where the B&B exact method of LINGO software takes longer to discover, and by limiting the execution time to 5 hours, LINGO becomes unable in most cases to achieve a better solution than meta-heuristics. For the same test problem instance, the gap between the simulated annealing and genetic algorithm reached 9.47%. This large deviation may be justified by the high exploitation that characterized the simulated annealing algorithm.

Regarding the execution time performance measure, Figure 3 and Figure 4 shows that the simulating annealing algorithm outperforms highly GAMS software, and GA although the performance of our PC is less than the one used in (Sahin & Alpay, 2016).

Table 7. Comparison of SA results (Bouaziz & Lemouari, 2019) with those of Yelliz reported in (Sahin & Alpay, 2016)

PBM	GAMS		GA			SA			GAMS-SA	GA-SA
(P×M×W× C)	F	T(s)	F_Best	F_Avg	T(s)	F_Best	F_Avg	T(s)	Gap(%)	Gap(%)
p1 (4,4,4,2)	7*	1	7	7	-	7	7	0	0.00	0.00
p2 (5,4,5,2)	11*	2	11	11	-	11	11	0	0.00	0.00
p3 (6,5,5,2)	18*	3	18	18	-	18	18	0	0.00	0.00
p4 (10,7,4,2)	34*	9	34	34	-	34	34	1	0.00	0.00
p5 (10,7,6,3)	25*	138	25	25	-	25	25	2	0.00	0.00
p6 (12,8,6,3)	29*	342	29	29.66	33	29	29	3	0.00	2.22
p7 (12,8,7,3)	33**	>18000	33	34.33	39	33	33	4	0.00	3.87
p8 (15,10,6,3)	50**	>18000	49	49.33	40	49	49	4	2.00	0.67
p9 (15,10,6,4)	39**	>18000	37	38.66	42	35	35	5	10.26	9.47
p10 (20,10,6,3)	63**	>18000	60	62	49	60	60.66	11	3.71	2.16
p11 (20,10,6,4)	47**	>18000	47	49.66	54	45	46.33	8	1.42	6.70

Figure 3. SA vs. GAMS software (B&B) regarding the execution time measure

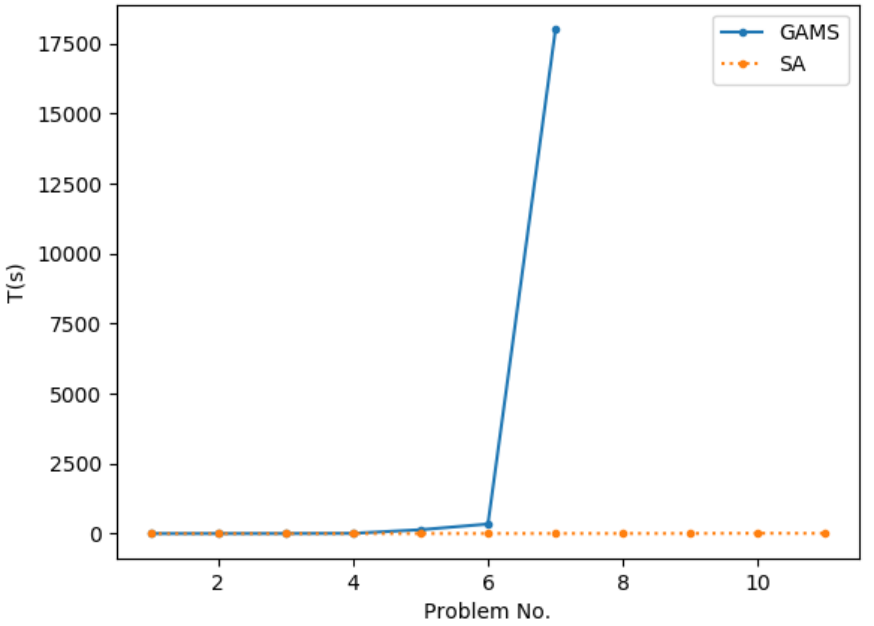
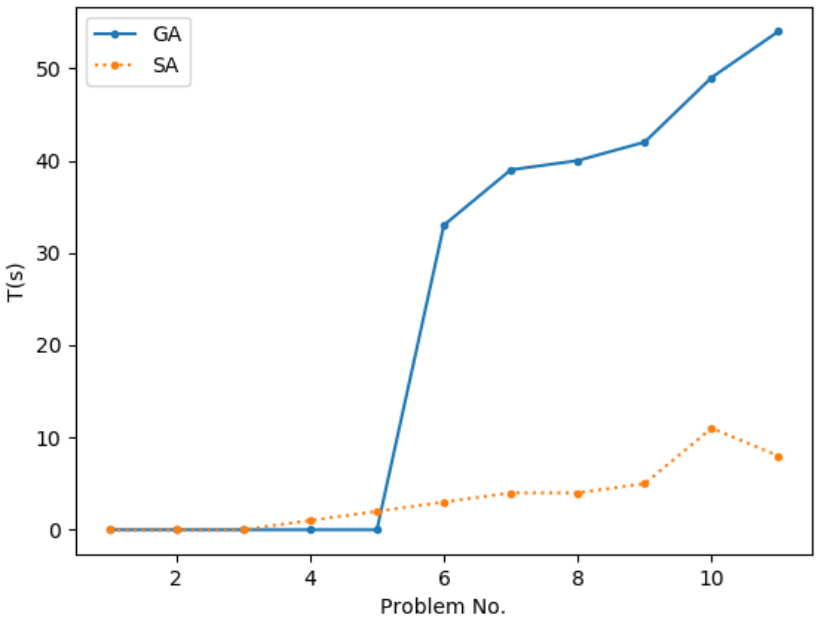


Figure 4. SA vs. GA regarding the execution time measure



CONCLUSION

In this paper, a variant of the cell formation problem, called Cubic Cell Formation Problem, has been addressed. A new strategy that combines the simulated annealing heuristics and a randomized search procedure is used to solve the problem. The proposed algorithm aims to find the compromise between the inter-cell movement of parts and workers, and the use of machines within cells. The compromise is controlled by the manufacturing system designer through a parameter which is associated to the heterogeneity objective. By associating different values to this parameter, several configurations may be obtained. For a best flexibility, the designer has the ability to fix the number of cells, or to let the algorithm decides on the number of cells resulting in the best objective values. The effectiveness of algorithm is tested on 11 test instances reached in the literature. The reported results show that the proposed algorithm gives the best results compared to the existed methods. The authors are looking forward to tackling the Generalized Cubic Cell Formation Problem (GCCFP) by considering the order between the machines used to process the parts, and considering the multiple processing routes for each part. In that case, more decisions must be taken to select the appropriate process plan for each part. Also, in future work, taking into account the machine reliability issue makes the problem more realistic. Concerning the resolution method, in this work, when solving the problem, the multiple objectives presented in the mathematical model are aggregated into a single objective function. The development of a multi-objective optimization technique to solve the problem would be considered in future works.

REFERENCES

Aalaei, A., & Shavazipour, B. (2013). The tchebycheff norm for ranking dmus in cellular manufacturing systems with assignment worker. *International Journal of Applied Operational Research*, *3*, 41–57.

Bootaki, B., Mahdavi, I., & Paydar, M. (2015). New bi-objective robust design based utilisation towards dynamic cell formation problem with fuzzy random demands. *International Journal of Computer Integrated Manufacturing*, *28*(6), 577–592. doi:10.1080/0951192X.2014.880949

Bootaki, B., Mahdavi, I., & Paydar, M. M. (2014). A hybrid GA-AUGMECON method to solve a cubic cell formation problem considering different worker skills. *Computers & Industrial Engineering*, *75*, 31–40. doi:10.1016/j.cie.2014.05.022

Bouaziz, H., & Lemouari, A. (2019). Solving the Cubic Cell Formation Problem Using Simulated Annealing Algorithm to Develop Multiple Configurations. *International Conference on Theoretical and Applicative Aspects of Computer Science (ICTAACS)*, 1-7. doi:10.1109/ICTAACS48474.2019.8988125

Bouaziz, H., & Lemouari, A. (2020). Bouaziz, H., Berghida, M., & Lemouari, A. (2020). Solving the Generalized Cubic Cell Formation Problem Using Discrete Flower Pollination Algorithm. *Expert Systems with Applications*, *150*, 113345. doi:10.1016/j.eswa.2020.113345

Danilovic, M., & Ilic, O. (2019). A novel hybrid algorithm for manufacturing cell formation problem. *Expert Systems with Applications*, *135*, 327–350. doi:10.1016/j.eswa.2019.06.019

Feng, H., Da, W., Xi, L., Pan, E., & Xia, T. (2017). Solving the integrated cell formation and worker assignment problem using particle swarm optimization and linear programming. *Computers & Industrial Engineering*, *110*, 126–137. doi:10.1016/j.cie.2017.05.038

Jiang, L., Ji, J., Lu, Y. Y. C., & Jia, Y. (2019). Mathematical modeling and simulated annealing algorithm for spatial layout problem. *Cluster Computing*, *22*(S3), 6383–6391. doi:10.1007/s10586-018-2137-8

Joines, J. A., King, R. E., & Culbreth, C. T. (1996). *A comprehensive review of production-oriented manufacturing cell formation techniques*. Academic Press.

Karoum, B., & Elbenani, Y. (2019). Discrete cuckoo search algorithm for solving the cell formation problem. *International Journal of Manufacturing Research*, *14*(3), 245–264. doi:10.1504/IJMR.2019.100991

Leite, N., Melício, F., & Rosa, A. (2019). A fast simulated annealing algorithm for the examination timetabling problem. *Expert Systems with Applications*, *122*, 137–151. doi:10.1016/j.eswa.2018.12.048

Mahdavi, I., Aalaei, A., Paydar, M. M., & Solimanpur, M. (2012). A new mathematical model for integrating all incidence matrices in multi-dimensional cellular manufacturing system. *Journal of Manufacturing Systems*, *31*(2), 214–223. doi:10.1016/j.jmsy.2011.07.007

Mahmoodian, V., Jabbarzadah, A., Rezazadeh, H., & Barzinpour, F. (2019). A novel intelligent particle swarm optimization algorithm for solving cell formation problem. *Neural Computing & Applications*, *31*(2), 801–815. doi:10.1007/s00521-017-3020-x

Min, H., & Shin, D. (1993). Simultaneous formation of machine and human cells in group technology: A multiple objective approach. *International Journal of Production Research*, *31*(10), 2307–2318. doi:10.1080/00207549308956859

Nikoofarid, E., & Aalaei, A. (2012). Production planning and worker assignment in a dynamic virtual cellular manufacturing system. *International Journal of Management Science and Engineering Management*, *7*(2), 89–95. doi:10.1080/17509653.2012.10671211

Nourie, H., Tang, S. H., Tuah, B. H., Ariffin, M. K., & Samin, R. (2013). Metaheuristic techniques on cell formation in cellular manufacturing system. *J Autom Control Eng*, *1*(1), 49–54. doi:10.12720/joace.1.1.49-54

Sahin, Y. B., & Alpay, S. (2016). A metaheuristic approach for a cubic cell formation problem. *Expert Systems with Applications*, *65*, 40–51. doi:10.1016/j.eswa.2016.08.034

Shiyas, C. R., & Pillai, V. M. (2014). A mathematical programming model for manufacturing cell formation to develop multiple configurations. *Journal of Manufacturing Systems*, *33*(1), 149–158. doi:10.1016/j.jmsy.2013.10.002

Tasoglubc, G., & Yildiz, G. (2019). Simulated annealing based simulation optimization method for solving integrated berth allocation and quay crane scheduling problems. *Simulation Modelling Practice and Theory*, *97*, 19–48.

Wu, T. H., Chung, S. H., & Chang, C. C. (2009). Hybrid simulated annealing algorithm with mutation operator to the cell formation problem with alternative process routings. *Expert Systems with Applications, 36*(2).

Hamida Bouaziz received her PhD in computer science from the University of Bourgogne Franche-Comté, France. She is currently an associate professor at the University of Jijel, Algeria. She is a member of the mechatronics laboratory of the same university. Her research interests include the use of formal methods in the specification and verification of complex systems, model-driven engineering (MDE) and model transformation, operational research, mathematical modeling, and optimization.

Ali Lemouari received his HDR (post-doctoral degree allowing its holder to supervise PhD students) degree and PhD degree in computer science from the university of Constantine2, Algeria. Currently, he is an associate professor at university of Jijel, Algeria. He is a member of LMAM Laboratory in the same university. His area of research includes operational research, mathematical modelling, and optimization.