



A multi-objective vibration damping optimization algorithm for solving a cellular manufacturing system with manpower and tool allocation

N. Aghajani-Delavar^a, E. Mehdizadeh^{a,*}, R. Tavakkoli-Moghaddam^b, and H. Haleh^c

a. Department of Industrial Engineering, Faculty of Industrial and Mechanical Engineering, Qazvin Branch, Islamic Azad University, P.O. Box 34185/1416, Qazvin, Iran.

b. School of Industrial Engineering, College of Engineering, University of Tehran, Tehran, Iran.

c. Department of Industrial Engineering, Golpayegan University of Technology, Golpayegan, Iran.

Received 7 December 2018; received in revised form 10 June 2020; accepted 26 September 2020

KEYWORDS

Dynamic cellular manufacturing system;
Tool allocation;
Multi-objective;
Vibration damping algorithm.

Abstract. In this paper, a novel bi-objective mathematical model is proposed to design a four-dimensional (i.e., part, machine, operator, and tool) Cellular Manufacturing System (CMS) in a dynamic environment. The main objectives of this model are to: (1) Minimize total costs including tools processing cost, costs of transporting cells between various cells, machine setup cost, and operators' educational costs and (2) Maximize the skill level of operators. The developed model is strictly NP-hard and exact algorithms cannot find globally optimal solutions in a reasonably computational amount of time. Thus, a Multi-Objective Vibration Damping Optimization (MOVDO) algorithm with a new solution structure that satisfies all the constraints and generates feasible solutions is proposed to find near-optimal solutions in a reasonably computational amount of time. Since there is no benchmark available in the literature, three other meta-heuristic algorithms (i.e., Non-dominated Sorting Genetic Algorithm-II (NSGA-II), Multi-Objective Particle Swarm Optimization (MOPSO), and Multi-Objective Invasive Weeds Optimization (MOIWO)) with a similar solution structure are developed to validate the performance of the proposed MOVDO algorithm for solving various instances of the developed model. The result of comparing their performances based on statistical tests and different measuring metrics reveals that the proposed MOVDO algorithm remarkably outperforms other meta-heuristics used in this paper.

© 2022 Sharif University of Technology. All rights reserved.

1. Introduction

Growing competition and diversity of products supplied by various competitors have persuaded companies

to use different efficient and new strategies on their procurement, production, and distribution sections. A significant proportion of the costs at each industrial center are related to its manufacturing system.

The most significant objectives of each manufacturing system are as follows: (a) use the capacity of different facilities more efficiently, (b) increase the productivity of workforce involved in performing different operations, (c) employ automated systems to increase the quality and production rate, (d) decrease the inventory level, and (e) reduce the material

*. Corresponding author.

E-mail addresses: aghajani.nacem@gmail.com (N. Aghajani-Delavar); emehdi@qiau.ac.ir (E. Mehdizadeh); tavakkoli@ut.ac.ir (R. Tavakkoli-Moghaddam); hhaleh@gut.ac.ir (H. Haleh)

transportation volume and increase the flexibility. A Cellular Manufacturing System (CMS) is one of the different types of manufacturing systems that can be used to achieve these goals.

CMSs are one of the most efficient alternatives for establishing systems with high diversity and production rates. The main objective of CMSs is to minimize materials' flow costs. One of the main crucial steps of designing these systems is to generate various cells in a productive environment. Each cell involves a family of parts with similar operations. Therefore, the machines are assigned to each cell to perform similar operations on each part. It is not a simple and efficient method to handle all the processes of manufacturing products into a single cell since the process of manufacturing special types of products should be performed into different segregated shops. These facts reveal the real need for designing different cells into a manufacturing system and assign specialized machines, operations, and workforces to each cell to reduce inter-cellular transportation.

A novel mixed-integer programming model is proposed in this paper to consider four different features of machines, parts, tools, and manpower in a Dynamic CMS (DCMS). The main purpose of this model is to find the most appropriate decisions to the following variables so that the best possible values of conflicting total tool processing cost minimization and skill level maximization objectives are obtained in a reasonable amount of computational time:

- Establishing a number of cells into different locations of a CMS;
- Locating some types of machines in the cells;
- Operationalizing a number of operators to perform different types of operations assigned to each machine;
- Allocating parts and tools to each machine;
- Determining the optimal skill level of operators that are going to perform various operations of the machines located in each cell;
- Defining part transmissions between the machines of each cell and those parts should be transferred among different cells.

Also, major assumptions including machine capacity, impracticality of inserting more than one specific tool on each box, impossibility of assigning more than one person to accomplish a job, and lifetime limitation of tools are considered to bring the model closer to real-world conditions.

The rest of this paper is organized as follows. The literature of the models and developed solution algorithms are discussed in the next section. The problem statement and the developed model for DCMS

are discussed in Section 3. The proposed solution algorithms to solve the models for small and large instances are discussed in Section 4. The presented metrics for evaluating the performance of multi-objective meta-heuristic algorithms and the used method for calibrating the main parameters of the developed algorithms are discussed in Section 5. The results and statistical tests to compare the performance of the developed algorithms are presented in Section 6. Finally, this paper is concluded in Section 7.

2. Literature review

DCMS problems have attracted the attention of various authors in recent years. Different mathematical models have developed in the literature to formulate this problem. Also, various solution algorithms are developed to solve different versions of this problem. So, this section is divided into three sections to investigate the main features of the developed models along with different types of algorithms proposed to solve the models.

2.1. Cell Formation Problem (CFP)

The most fundamental decision that should be taken to establish an efficient CMS is the CFP. This is the reason why the motivated researchers attempt to define different formations and solution algorithms to solve this problem. Sofianopoulou [1] developed a conceptual model for embedding a group of machines and designing a medium-sized CMS. The model aims to minimize handling costs of inter-cellular material movements. A novel version of the Simulated Annealing (SA) algorithm was proposed to solve medium- and large-scale instances of the proposed model. Guerrero et al. [2] presented a novel two-stage structure of a DCMS. To do so, similar parts and machines were located on their pertinent families and cells, respectively. A novel self-organized neural network approach was implemented to divide similar parts into each group. Then, a novel linear model was proposed to assign the machines to their related part family. Finally, a novel heuristic algorithm was developed based on the main concepts and structures of the maximum spanning tree approach for solving benchmark examples of the proposed model. Soleymanpour et al. [3] presented a standard version of a CFP. Some supplementary approaches were embedded into the main mechanism of an effective neural network approach to solve 18 various benchmark instances. The proposed approach could obtain high-quality solutions with a shorter computational time.

Logendran and Karim [4] presented a novel non-linear programming model for dynamic CFP. Two major features of a low distance between active locations and employing various vehicles with a limited capacity to transport commodities among different cells were considered into the main process of designing a CMS.

A novel Tabu Search (TS) algorithm was proposed to solve the model's different instances. Spiliopoulos and Sofianopoulou [5] proposed a three-step method for designing a DCMS. The main contribution of this approach was to consider all the complicated computational aspects involved in the process of designing a flexible manufacturing system. A novel version of the TS algorithm was proposed to solve various test problems. Also, a novel searching strategy along with a proper combination of short- and long-term memories was designed into the main structure of the TS algorithm to enhance solution qualities and reduce the amount of time required for obtaining the best possible solutions. Prabhakaran et al. [6] developed a novel formulation for a DCMS. The model aimed to categorize machines and parts into various groups and cells so that the volume of inter-cellular movements and cell load variation were minimized. They proposed a novel ant colony optimization algorithm to solve small- and large-sized instances of the proposed formulation.

Defersha and Chen [7] developed a comprehensive mathematical model for DCMS and considered two significant features of tools availability and satisfaction of machines' needs for tools requirements in the procedure of designing a SCMS. The above model aimed to minimize total costs including setup costs, tools consumption costs, and operational costs so that various constraints including cell size limitations, machine capacity, and machine adjacency constraints would be completely satisfied. Saidi-Mehrabad and Safaei [8] developed a novel mathematical model for the SCMS. The main purpose of designing this system was to classify parts and machines into different groups and cells so that the total material handling costs could be minimized. A novel neural network method was proposed to solve the different test problems associated with the model. Defersha and Chen [9] proposed a novel model for a CMS and lot sizing in a dynamic environment. The main purpose of developing this model was to investigate the effect of lot-sizing on product quality. The model aimed to minimize production and quality-related costs. A novel version of Genetic Algorithm (GA) was proposed to solve the model.

Tavakkoli-Moghaddam et al. [10] proposed a novel bi-objective model for a CMS in a dynamic and multi-product environment. An efficient version of the SA algorithm was proposed to solve small- and large-scale instances. An exact algorithm was employed to find optimal solutions to various test problems. The gap between the best global solutions and solutions of the proposed SA was less than 4%. Mahdavi et al. [11] proposed a novel mathematical model for a hybrid cell formation and cell layout problems. Similar parts and machines were classified into their pertinent groups and cells to enhance the capability of the system to

fulfill customer requirements. New performance and neighboring criteria were employed to evaluate the quality of the obtained solutions.

Majazi Dalfard [12] presented a new nonlinear integer programming model for dynamic CFP in a CMS. The main contribution of their model was to enhance the capability of a manufacturing system to transport a large number of commodities among various cells. The main procedure of the SA algorithm was embedded in a branch-and-bound algorithm to obtain more qualified solutions. Paydar and Saidi-Mehrabad [13] developed a novel mathematical formulation for a dynamic CFP. The model aimed at ensuring the grouping efficiency of the entire system in the case that the number of cells required for generating different products be unknown. Two small test problems were presented to show the effectiveness of the proposed model. A combination of genetic and variable neighborhood algorithms was developed to solve the model. Salarian et al. [14] developed a novel stochastic formulation for a dynamic manufacturing system. Normal distribution was employed to consider the uncertainty of demand and processing time. Also, an exponential distribution was employed to consider the parts' inter-arrival time. The main purpose of the model was to define a very efficient layout for machines located in each cell so that inter-cellular movements along with the number of bottlenecks were concurrently minimized. Numerical examples were provided to illustrate the efficiency of the proposed formulation.

Bychkov and Batsyn [15] proposed a novel mixed-integer linear model for a dynamic cell manufacturing problem. The model aimed to determine the optimal value of the famous grouping efficacy measure. Various test problems of the proposed formulation were implemented using CPLEX software and optimal solutions were obtained in a short time period. Zohrevand et al. [16] proposed a novel mathematical model for a DCMS to formulate two major concerns about human resource aspects and the stochastic nature of the model's main parameters. The model aimed to optimize two conflicting objectives of minimizing the total costs of a CMS and maximizing the productivity of the workforce considering for executing various jobs. A novel heuristic algorithm obtained by combining the main characteristics of both genetic and SA algorithms was proposed to solve small- and large-scale instances.

2.2. Resource assignment and CFP

Another important feature of cell manufacturing systems is to determine the workforce size to be assigned to different cells. Considering workforce assignment to the main structure of the CFP makes the solutions closer to real-world considerations. This problem has been considered in different research projects and the

main purpose of some recent papers was to propose a proper formulation for this problem or develop solution algorithms and obtain optimal or near-optimal solutions to small- and large-scale instances of this problem in a reasonable computational time. Mahdavi et al. [17] extended a two-dimensional machine-part matrix of dynamic CFP for a three-dimensional matrix to consider the assignment of workers to cells and bring the model closer to real conditions of manufacturing systems. The model aimed to minimize the number of voids in a three-dimensional machine-part-worker situation. The main assumption of the model was related to assigning each worker to various types of jobs. The solutions of various test problems were statistically compared to the solution obtained by previous research works to exhibit the efficiency of the proposed algorithm. Bagheri and Bashiri [18] proposed a novel hybrid model for a CFP, which minimizes inter-cellular movements, machine costs, and operator related issues at the same time. They used an LP-metric method to aggregate different objectives and turn the model into a single-objective one. A branch-and-bound algorithm was employed to find exact solutions of various test problems into a logical computational time.

Saidi-Mehrabad et al. [19] proposed a comprehensive mathematical model for dynamic production systems that integrated the production planning and worker training which aimed to minimize the overhead costs, maintenance, cell reconfiguration, inventory maintenance, number of deferred orders, training and workers' salary costs. Paydar et al. [20] proposed a novel formulation for a dynamic cell manufacturing system. The model aimed to minimize various cost components including maintenance and overhead costs, inventory costs, system reconfiguring costs, and workers' salary and training costs. The main objective of this model was to combine production planning, worker training, operation sequence, and multi-period planning horizon into an efficient manufacturing system. Various test problems were solved optimally to prove the efficiency of the proposed model. Mehdizadeh and Rahimi [21] proposed a novel mathematical model for a multi-objective CFP to find an optimal assignment of workforce, number and type of machines located in different cells, and all the locations, in which the cells could be established so that conflicting objectives of inter-cellular movement minimization, machine and operator cost minimization, and consecutive forward flow maximization objectives are simultaneously optimized. Since the model was extremely NP-hard, multi-objective SA and vibration damping optimization algorithms were developed to solve different test problems associated with the model. Mehdizadeh et al. [22] proposed a novel mixed-integer programming formulation for a bi-objective CFP. The most impor-

tant contribution of this paper was to set a resource limitation in designing the overall structure of a CMS and bring the model closer to real-world conditions. They proposed a novel Multi-Objective Vibration Damping Optimization (MOVDO) algorithm to solve their model. Also, two other multi-objective GAs were developed to validate the performance of the proposed algorithm in solving large-scale instances. Feng et al. [23] proposed a novel mathematical model for hybrid cell formation and resource assignment problems to define optimal assignment of machines, parts, and workers so that all the costs relating to different stages of establishing a CMS would be minimized. They developed a novel hybrid version of particle swarm optimization algorithm to solve their model.

Forghani and Fatemi Ghomi [24] formulated a mixed-integer nonlinear mathematical model to minimize the production, subcontract, material handling, machine idleness, and handling costs. To solve the model, a heuristic method is suggested. Hashemoghli et al. [25] presented a non-linear mixed-integer programming model under an uncertain environment to minimize the total costs and total inaction workers and machines, simultaneously. They employed GAMS to solve their model after linearizing it.

2.3. Solution algorithms developed for solving the DCMS

Different heuristic and meta-heuristic algorithms were proposed in the literature to find suitable solutions to complicated and NP-hard problems of dynamic manufacturing systems in a reasonable amount of time. Vakharia and Chang [26] extended TS and SA algorithms to propose better heuristic methods for solving a DCMS. The proposed heuristic algorithms were employed to solve benchmark examples of a real-world industry and obtain more qualified solutions. Computational results illustrated that the SA algorithm could obtain better results in terms of solution quality and computational time. Ravichandran and Chandra Sekhara Rao [27] presented a novel model for a dynamic CFP and implemented a novel fuzzy clustering approach to identify the best classification of parts into different families. Also, a novel similarity coefficient method was used to define different families and classify similar parts into each unique family. Numerical examples were proposed to explain the performance of the proposed formulation.

Lozano et al. [28] proposed two novel neural network methods of Hopfield and Potts mean field annealing algorithms for solving a CFP. The model aimed to minimize total transportation costs. The proposed quadratic formulation for a CFP could employ symmetric and sequential based similarity coefficients to identify distances of every two consecutive machines. Computational results demonstrated that

the developed Potts mean field annealing had a better performance in terms of both solution quality and computational time. Lozano et al. [29] proposed a novel fuzzy C-mean algorithm for solving a CFP with different part families and parallel machines. The model aimed to minimize total inter-cellular movements and the number of voids. Comparing obtained solutions with other solutions reported in the literature revealed the capability of this method to generate high-quality solutions. Wu et al. [30] developed a novel version of the TS algorithm to solve small- and medium-sized instances of the proposed formulation. They embedded a random approach along with the group and assigned the method to the main structure of their algorithm to generate more appropriate initial solutions. The proposed algorithm could find the model's optimal solutions in a very short period.

Diaby and Nsakanda [31] developed a heuristic method based on the Lagrange relaxation method to create near-optimal solutions in large-scale CMSs. Demands of parts would be satisfied through internal production or outsourcing. The goal was to minimize handling, manufacturing, outsourcing, and setup costs. Lei and Wu [32] proposed a novel multi-objective TS algorithm for solving a CFP. All the parts and machines were categorized into different families and groups, respectively. The first objective aimed to minimize intra-movements and inter-movements. The second objective aimed to minimize total cell variations. A GA was developed to evaluate the performance of the proposed algorithm in solving various test problems. Computational results demonstrated that the proposed algorithm could obtain better optimal Pareto solutions.

Wu et al. [33] developed a hierarchical GA for solving a CFP to determine the cell and machine layouts of a manufacturing cell system. The proposed manufacturing system was mainly designed to determine three major aspects of cell formation, group layout, and group scheduling in a dynamic environment. A hierarchical procedure was proposed to design a correlated fitness function along with a novel mutation operator into the main structure of the GA to elaborate on the algorithm's capability to seek new feasible solution regions and obtain a better solution. Computational results demonstrated that the proposed algorithm could obtain high-quality solutions. Mukattash et al. [34] developed three heuristic methods to solve complicated instances of a CFP. The main purpose of presenting their methods was to identify the best state of assigning parts to cells so that three major concerns about the location of parallel machines, processing times, and processing plans could be taken into consideration.

James et al. [35] proposed a novel grouping GA to solve a CFP. They embedded a local search procedure into the main structure of the GA to en-

hance the algorithm performance in obtaining better solutions. Statistical comparisons revealed that the proposed algorithm could obtain for all small- and large-scale test problems. Ghotboddini et al. [36] implemented a Bender's decomposition algorithm on GAMS software to solve various instances of a mixed-integer linear programming formulation for a CFP. The first objective of the proposed model aimed to minimize overtime costs of employing equipment and workforce along with reassignment costs of workforce assigned to various jobs. The second objective is mainly designed to maximize the workforce' utilization rate. Martins et al. [37] developed a novel version of Variable Neighborhood Search (VNS) algorithm to solve complicated CFP instances. A random ordering-based local search procedure was embedded into the main mechanism of the proposed algorithm to discover more feasible regions and obtain better solutions.

Kao and Li [38] proposed a novel clustering algorithm to solve various instances of a CFP. The main concept of the algorithm was inspired by random meeting of ants to generate initial clusters. Also, two other living features of ants including randomization and collective behavior enable algorithms to modify wrongly classified parts and bring them to a better state. Rafiei and Ghodsi [39] proposed a novel ant colony optimization algorithm to solve NP-hard instances of a bi-objective CFP. Also, a mutation operator of the GA was added to the main procedure of the proposed algorithm to elaborate diversification characteristics of the proposed algorithm. Zeidi et al. [40] presented a novel hybrid metaheuristic algorithm to solve intricate examples of a nonlinear mixed-integer programming model of a dynamic CFP. A specific procedure of the neural network method was implemented into the main mechanism of the GA to solve various instances of this problem and obtain more qualified near-optimal solutions.

Shiyas and Madhusudanan Pillai [41] proposed a novel hybrid version of the GA to solve CFP. The model aimed to optimize two conflicting objectives of intercellular movements and cell heterogeneity at the same time. The proposed algorithm was able to find appropriate near-optimal solutions in a shorter computational time. Rezazadeh et al. [42] proposed a novel particle swarm optimization algorithm for solving a dynamic CFP. The main purpose of developing a novel formulation for a CFP was to identify the optimal number of cells located in a dynamic manufacturing system so that manufacturing, material handling, subtracting, inventory, and production costs could be minimized. The local search mechanism of SA was embedded into the main structure of particle swarm optimization to generate better solutions. Buruk Sahin and Alpay [43] proposed a novel GA for solving NP-hard instances of a CFP. A Taguchi method was implemented to

calibrate the main parameters of the proposed algorithm. Computational results demonstrated that the proposed algorithm could find qualified solutions for small- and large-scale benchmark instances of a CFP. Durga Rajesh et al. [44] proposed a novel similarity coefficient-based methodology to determine the main formation of cells located in a dynamic manufacturing system. This method could eliminate the drawbacks of previously proposed methods and obtain more appropriate solutions. Also, major criteria of a CMS were considered to evaluate the performance of their developed method in solving small- and large-sized instances of the problem.

Behnia et al. [45] formulated a multi-dimensional CMS to minimize the total number of voids and balance the workload assigned to cells. To solve the model, two Nested Bi-level metaheuristics were implemented. Rostami et al. [46] formulated a dynamic virtual CFP to maximize the total profits of the factory, the grouping efficacy, and the number of the new product. Multi-choice goal programming with a utility function is used to solve the model after linearization. Tavanayi et al. [47] formulated a nonlinear mixed-integer programming model for the cooperative CMS problem to minimize the total costs of inter-cell and intra-cell material handling and intra-factory material handling. The cooperative game theory method to solve the problem of reducing costs between different companies.

A simple glimpse at the main features of the papers discussed in Table 1 reveals that simultaneous consideration of machine, parts, resources, and tools into the main process of presenting a mathematical formulation for dynamic manufacturing systems has not been investigated yet. So, the main focus of this paper is to propose a novel mathematical model for a dynamic manufacturing system while machines, tools, parts, and resources are concurrently considered into the main structure of the proposed formulation to bring the results closer to real-world assumptions. In this paper, a new mathematical model in a DCMS is proposed considering the skill level of operators and tools as the third and fourth dimensions simultaneously. Because cell formation and simultaneous resource allocation are considered, it is an integrated CMS that is closer to reality. In the case of the solution method, fuzzy logic is used to identify non-dominated solutions and provide an efficient solution representation for the proposed problem.

3. Problem formulation

Our proposed problem is a DCMS in which the number of P -parts, M -machines, O -operations, L -tools, W -operators, C -cells, and a set of S -skill levels are available. In this case, each part needs a set of operations and each operation can be performed by a

set of machines, tools, and operators. Each operator has different skill levels for each operation with a variable cost; each machine is placed in only one cell; and movement between cells has a cost.

3.1. Problem statement

The main purpose of developing a bi-objective mixed-integer model for a DCMS is to:

1. Assign a specific tool to each machine;
2. Allocate each machine to several types of tools;
3. Determine the operator that should be assigned for processing a part on each machine;
4. Classify operators in various skill levels.

The main assumptions considered for developing a novel bi-objective model for the DCMS are given below:

- The time required for the operator to process each part on all the machines is known;
- The machine capacity to produce semi-product materials is known and limited;
- Several similar machines are used to respond to capacity requirements;
- Only one operator can be allocated for processing a part on each machine;
- It is not permitted to put repetitive tools on each tools box;
- Assigning each tool to more than one machine is not permitted;
- Life time of each tool is limited;
- Operators are classified at various skill levels;
- Each machine is allowed to be assigned to more than one tool at the same time.

3.2. Mathematical model

Indices, parameters, decision variables, objectives, and constraints of the novel bi-objective mixed-integer programming model proposed for the DCMS are as follows:

Indices

p	Index for parts ($p = 1, \dots, P$)
w	Index for operators ($w = 1, \dots, W$)
m	Index for machines ($m = 1, \dots, M$)
o	Index for operation ($o = 1, \dots, O$)
c	Index for cells ($c = 1, \dots, C$)
l	Index for tools ($l = 1, \dots, L$)
s	Index for skill levels ($s \in \{\text{low, medium, high}\}$)

Table 1. Summary of recent studies on the DCMS.

Resources	Objective function				Solution approach													
	O1	O2	O3	O4	MP	H	TS	SA	A	GA	VNS	PSO	SS	S	F	N	VDO	
Defersha and Chen [7]		✓		✓						✓								
Saidi-Mehrabad and Safaei [8]	✓				✓											✓		
Defersha and Chen [9]	✓	✓			✓					✓								
Tavakkoli-Moghaddam et al. [10]	✓							✓										
Mahdavi et al. [17]			✓											✓				
Bagheri and Bashiri [18]	✓	✓		✓	✓													
Saidi-Mehrabad et al. [19]	✓			✓	✓													
Mahdavi et al. [11]			✓			✓												
Majazi Dalfard [12]						✓		✓										
Salarian et al. [14]	✓													✓				
Bychkov and Batsyn [15]			✓		✓													
Zohrevand et al. [16]		✓	✓		✓	✓		✓		✓								
Ghotboddini et al. [36]		✓	✓		✓													
Martins et al. [37]	✓										✓							
Rafiei and Ghodsi [39]	✓								✓	✓								
Zeidi et al. [40]			✓							✓						✓		
Shiyas and Madhusudanan Pillai [41]	✓									✓								
Rezazadeh et al. [42]		✓										✓						
Buruk Sahin and Alpay [43]	✓									✓								
Durga Rajesh et al. [44]			✓											✓				
Hajipour et al. [48]			✓														✓	
Tavakkoli-Moghaddam et al. [49]		✓	✓														✓	
Hajipour et al. [50]		✓		✓													✓	
Hajipour et al. [51]	✓	✓															✓	

Notes I: Major objectives considered by different models (shown in column 2).

O1: Maximizing cell independence (minimizing inter-cellular movements);

O2: Considering costs (e.g., machine operating cost, machine modification costs, machine reconfiguration cost, and machine setup cost);

O3: Considering machine utilization;

O4: Considering operators costs (e.g., operator training cost and operator allocation cost);

Note II: Solution approaches used (shown in column 3).

MP: Mathematical Programming model; H: Heuristics;

TS: Tabu Search; SA: Simulated Annealing; A: Ant colony optimization; GA: Genetic Algorithms;

VNS: Variable Neighborhood Search; PSO: Particle Swarm Optimization; SS: Scatter Search;

S: Simulation; F: Fuzzy theory; N: Neural networks.

Parameters

AB_{mw}	1 if operator w can perform specific operations of type m machine; 0, otherwise
am_{om}	1 if machine type m can perform type o operation; 0, otherwise
ap_{op}	1 if type o operation can be performed on part p ; 0, otherwise
E_{pi}	1 if tool l can be implemented for processing part p ; 0, otherwise
LM_c	Minimum number of machines that should be located in cell c
LP_c	Minimum number of parts processed by machines of cell c
LW_c	Minimum number of operators assigned for processing various parts on the machines, which are located in cell c
NM_m	Number of available type m machines located in various cells
NL_l	Number of available type l tools used for performing various operations of the machines located in different cells
NW_w	Number of available type w operators
T_{pmlwsc}	Time of processing part p on type m machine into cell c by means of operator w with skill levels and tools l
C_{pmlwsc}	Cost of processing part p on type m machine into cell c by means of operator w with skill levels and tools l
Q_{pomws}	Quality level of operator w with skill levels to perform operation type o by machine type m on part p
$CH_{pcc'}$	Costs of transporting part p between cells c and c'
$TH_{pcc'}$	Time required for transporting part p between cells c and c'
KM_m	All the time that machine m is available
KW_w	Time required to change the cell that operator w is assigned to it
HP_p	Cost required for changing the position of part p into the cell
HL_l	Processing costs of tools l into each cell
SU_{pom}	Setup cost required for performing operation o on tool p of machine m
B_{mw}	Learning costs of operator w to perform operations on machine m
TS_l	Number of sluts that tools l may occupy into tools box
MC_m	Maximum number of tools assigned to machine m

Decision variables

I_{pc}	1 if part p is assigned to cell c ; 0, otherwise
a_{pom}	1 if operation o of part p is required to be performed by machine m ; 0, otherwise
J_{poc}	1 if operation o of part p is processed in cell c ; 0, otherwise
X_{pmlwsc}	1 if operator w with skill level s and tool l processes part p on machine m into cell c ; 0, otherwise
$WH_{pocc'}$	1 if part p is transported from cells c to c' when operation o is completely processed on it; 0, otherwise
Z_{wc}	1 if operator w is assigned to cell c ; 0, otherwise
TR_{mw}	1 if operator w is trained to work on machine m ; 0, otherwise
V_{ml}	1 if tools l is assigned to machine m ; 0, otherwise
G_{pom}	1 if operation o of part p is performed on machine m ; 0, otherwise
DL_l	1 if tool l is selected to work on a specific machine; 0, otherwise
DM_m	1 if type m machine is selected to be assigned to a specific cell; 0, otherwise
JM_{mc}	Number of type m machines that should be assigned to cell c
JL_{lc}	Number of type l tools that should be assigned to cell c
F_{pomws}	1 if operator w with skill level s is required to process the o th operation of part p on machine m ; 0, otherwise

Objective functions and constraints

$$\begin{aligned}
 \min Z1 = & \sum_{p=1}^P \sum_{m=1}^M \sum_{l=1}^L \sum_{w=1}^W \sum_{s=1}^S \sum_{c=1}^C \\
 & T_{pmlwsc} C_{pmlwsc} X_{pmlwsc} \\
 & + \sum_{p=1}^P \left(\sum_{c=1}^C I_{pc} - 1 \right) HP_p \\
 & + \sum_{p=1}^P \sum_{c=1}^C \sum_{c'=1}^C CH_{pcc'} TH_{pcc'} \\
 & \left(\sum_{o=1}^O WH_{pocc'} \right) + \sum_{p=1}^P \sum_{o=1}^O \sum_{m=1}^M SU_{pom} G_{pom} \\
 & + \sum_{w=1}^W \sum_{m=1}^M B_{mw} TR_{mw}, \quad (1)
 \end{aligned}$$

$$\max Z2 = \frac{\sum_{p=1}^P \sum_{o=1}^O \sum_{m=1}^M \sum_{w=1}^W \sum_{s=1}^S Q_{pomws} F_{pomws}}{\sum_{w=1}^W NW_w}, \quad (2)$$

s.t.

$$\sum_{l=1}^L \sum_{c=1}^C X_{pmlwsc} \leq DM_m \quad \forall p, m, w, s, \quad (3)$$

$$\sum_{m=1}^M \sum_{c=1}^C X_{pmlwsc} \leq DL_l \quad \forall p, l, w, s, \quad (4)$$

$$\sum_{c=1}^C X_{pmlwsc} \leq a_{pom} AB_{mw} \quad \forall p, m, l, w, s, \quad (5)$$

$$\sum_{m=1}^M JM_{mc} \geq LM_c \quad \forall c, \quad (6)$$

$$\sum_{p=1}^P I_{pc} \geq LP_c \quad \forall c, \quad (7)$$

$$\sum_{w=1}^W Z_{wc} \geq LW_c \quad \forall c, \quad (8)$$

$$\sum_{c=1}^C JM_{mc} \leq NM_m \quad \forall m, \quad (9)$$

$$\sum_{c=1}^C JL_{lc} \leq NL_l \quad \forall l, \quad (10)$$

$$\sum_{m=1}^M TR_{mw} \leq 1 \quad \forall w, \quad (11)$$

$$\sum_{c=1}^C \sum_{w=1}^W \sum_{l=1}^L X_{pmlwsc} \leq DM_m \quad \forall p, m, s, \quad (12)$$

$$\sum_{c=1}^C \sum_{w=1}^W \sum_{l=1}^L \sum_{p=1}^P \sum_{s=1}^S X_{pmlwsc} \geq DM_m \quad \forall m, \quad (13)$$

$$\sum_{m=1}^M V_{ml} \leq 1 \quad \forall l, \quad (14)$$

$$\sum_{l=1}^L V_{ml} \geq DM_m \quad \forall m, \quad (15)$$

$$\sum_{c=1}^C \sum_{w=1}^W X_{pmlwsc} \leq V_{ml} \quad \forall p, m, l, s, \quad (16)$$

$$\sum_{l=1}^L TS_l V_{ml} \leq MC_m \quad \forall m, \quad (17)$$

$$I_{pc} = \min \left(1, \sum_{m=1}^M \sum_{w=1}^W \sum_{s=1}^S \sum_{l=1}^L X_{pmlwsc} \right) \quad \forall p, c, \quad (18)$$

$$\sum_{c=1}^C I_{pc} \leq 1 \quad \forall p, \quad (19)$$

$$a_{pom} + 1 \geq am_{om} + ap_{op} \quad \forall p, o, m, \quad (20)$$

$$J_{poc} + J_{p(o+1)c'} < 1 + WH_{pocc'} \quad \forall p, o, c, c', \quad (21)$$

$$I_{pc} \in \{0, 1\} \quad \forall p, c, \quad (22)$$

$$I_{poc} \in \{0, 1\} \quad \forall p, o, c, \quad (23)$$

$$a_{pom} \in \{0, 1\} \quad \forall p, o, m, \quad (24)$$

$$WH_{pocc'} \in \{0, 1\} \quad \forall p, o, c, c', \quad (25)$$

$$X_{pmlwsc} \in \{0, 1\} \quad \forall p, m, l, w, s, c, \quad (26)$$

$$Z_{wc} \in \{0, 1\} \quad \forall w, c, \quad (27)$$

$$TR_{mw} \in \{0, 1\} \quad \forall m, w, \quad (28)$$

$$DM_m \in \{0, 1\} \quad \forall m, \quad (29)$$

$$DL_l \in \{0, 1\} \quad \forall l, \quad (30)$$

$$V_{ml} \in \{0, 1\} \quad \forall m, l, \quad (31)$$

$$G_{pom} \in \{0, 1\} \quad \forall p, o, m, \quad (32)$$

$$F_{pomws} \in \{0, 1\} \quad \forall p, o, m, w, s, \quad (33)$$

$$JM_{mc}, JL_{lc} \geq 0 \text{ and is intger} \quad \forall m, l, c. \quad (34)$$

The first objective function (1) includes five different components of minimizing tools processing costs, intra-cellular transportation costs, inter-cellular transportation costs, the machines' setup costs, and operators' educational costs. The second objective function (2) is mainly designed to maximize operators' skill level. Constraint (3) determines whether or not a machine can be employed in a particular period to produce various parts. Constraint (4) determines whether a tool is used at a time interval of manufacturing various parts or not. Constraint (5) ensures that only one operator can be assigned for processing a part on each machine. Constraint (6) determines if the minimum number of machines can be assigned to each cell. Constraint (7) identifies the minimum number of parts that can be assigned to each cell. Constraint (8) identify a

lower bound for the number of operators that can be allocated to each cell. Constraint (9) ensures that the number of machines on each type assigned to various cells should not go beyond a predetermined higher bound. Constraint (10) ensures that a number of the tools on each type should not be more than a predetermined upper bound. Constraint (11) ensures that each operator can be trained to work on only one type of machine available on different cells. Constraint (12) ensures that machine preparation should be performed before starting different types of operations. In other words, this constraint ensures the readiness to operate when processing various types of operations. Constraint (13) ensures that at least one operation should be executed when performing setup activities of that machine has been previously completed. Constraint (14) implies that each tool can be assigned to more than one machine. Constraint (15) implies that each machine can be assigned to several tools at the same time. Constraint (16) ensures the satisfaction of the relations between two types of decision variables. Constraint (17) ensures the capacity satisfaction of the tools box assigned to each machine. Constraints (18) and (19) are mainly used to guarantee the processing of part p into cell c . Constraint (20) implies that an operation can be assigned to each part and machine if processing that operation on the part is needed and the machine can process the operation on the part. Constraint (21) implies that inter-cellular transportation should be performed when two consecutive jobs are simultaneously performed on two various cells. Constraints (22)–(33) are mainly used to identify various binary decision variables of the model. Finally, Constraint (34) is mainly used to define the positive and integer variables of the model.

Each objective function is solved without considering the other objective function and after solving it, the other objective function solution is obtained. Given that the first objective function is to minimize costs and the second objective function is to maximize quality, the conflict between the objectives is shown in Figure 1.

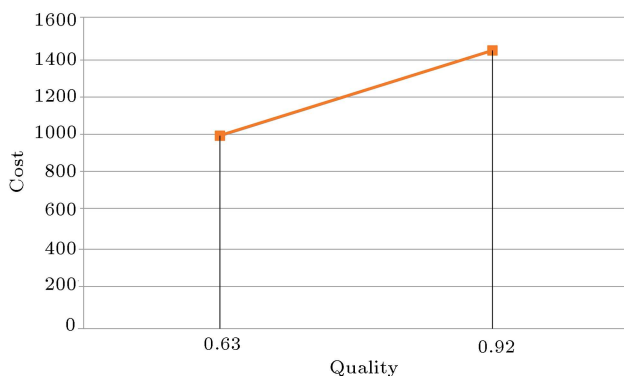


Figure 1. Conflict of the objectives.

4. Solution method

4.1. Solution representation

A novel solution structure is presented in this section to ensure the satisfaction of all the constraints and generate feasible solutions. This structure includes $c+6$ rows and $c+w+m$ columns, where c , w , and m are the number of cells, number of operators, and number of machines, respectively. All the numbers are randomly generated between 0 and 1. The numbers shown in various vectors and columns of this structure are not significant. So, a decoding procedure is developed into six different steps to satisfy all the constraints and determine the main purpose of using these numbers. A small example with two cells, three machines, six tools, four parts, and four operators is presented in this section to demonstrate the procedure of satisfying constraints. According to this example, all the tools and machines can process different types of operations on the parts, which are assigned to different cells based on specific types of operations needed to be performed on their physical structure. According to the model's main assumptions, the solution structure of this example is shown in eight vectors and 13 columns, as shown in Figure 2.

In the first section of the decoding structure, we need to consider the first c rows of solution structure for determining the best possible assignment of machines, tools, and operators. The first step of this section begins by considering a counter with zero initial value and adding one unit to it when a machine is assigned to each cell. Therefore, the last value of this counter equals three since the example considered in this section includes three machines.


Firstly, the first three elements of these vectors will be separated. Then, numbers that are higher than 0.5 will be replaced with 1 and the other numbers will be replaced with zero. In the case that all the numbers are lower than 0.5, the biggest number will be replaced with one and the others will be changed to zero. This process is shown in Figure 3. This figure shows that Machine 1 should be assigned to cell 1 and the other two cells should be assigned to cell 2.

The second step of this section concerns assigning various operators to each cell. To this end, the numbers located between $counter+1$ and $counter+w$ of vectors 1 and 2 will be separated. We assume that the rows and columns of separated numbers are indicators of cells and operators. So, the numbers less than 0.5 will be replaced with 0 while the other numbers will be replaced with 1. If no operator is assigned to one cell, the largest number will be replaced with one and the other numbers will be equal to zero. This process ensures the fact that at least one operator should be assigned to each cell. The process of operators' assignment is schematically shown in Figure 4.

Machine, tool and operator assignment to cell	0.55	0.35	0.11	0.25	0.61	0.78	0.13	0.31	0.17	0.92	0.54	0.67	0.26
	0.45	0.74	0.98	0.42	0.35	0.87	0.91	0.64	0.42	0.73	0.37	0.8	0.1
Weight	0.35	0.31	0.08	0.35	0.41	0.87	0.27	0.15	0.82	0.6	0.52	0.47	0.38
Part assignment	0.58	0.6	0.26	0.65	0.68	0.74	0.45	0.14	0.98	0.29	0.07	0.54	0.32
Cell assignment	0.15	0.19	0.41	0.75	0.82	0.79	0.32	0.71	0.62	0.67	0.71	0.82	0.47
Tool assignment	0.49	0.8	0.14	0.42	0.92	0.79	0.96	0.47	0.24	0.39	0.31	0.84	0.74
Machine assignment	0.03	0.44	0.38	0.77	0.80	0.19	0.49	0.45	0.65	0.71	0.75	0.28	0.68
Operator assignment	0.66	0.16	0.12	0.50	0.96	0.34	0.59	0.22	0.75	0.26	0.51	0.70	0.89

Figure 2. Representation of the solution structure.


	M1	M2	M3
Cell1	0.5	0.3	0.1
Cell2	0.4	0.7	0.9



	M1	M2	M3
Cell1	1	0	0
Cell2	0	1	1

Figure 3. Process of assigning machines to different cells.

	W1	W2	W3	W4
Cell1	0.2	0.6	0.7	0.1
Cell2	0.4	0.3	0.8	0.9



	W1	W2	W3	W4
Cell1	0	1	1	0
Cell2	0	0	1	1

Figure 4. Schematic representation of assigning operators to the cells.

Cell 1	0.3	0.1	0.9	0.5	0.6	0.2
Cell 2	0.6	0.4	0.7	0.3	0.8	0.1
Weight	0.1	0.8	0.6	0.5	0.4	0.3

ANL	5	4	8	5	6	4
Ntool	1	4	5	3	3	2
Cell 1	0.33	0.20	0.56	0.63	0.43	0.67
Cell 2	0.67	0.80	0.44	0.38	0.57	0.33

	Tool 1	Tool 2	Tool 3	Tool 4	Tool 5	Tool 6
Cell 1	1	1	3	2	2	2
Cell 2	1	4	3	2	2	1

Figure 5. Schematic representation of allocating tools to cells.

The last step of this section is related to tools assignment. This step begins with separating the last $counter + w + 1$ element of the first $c + 1$ rows. Then, a process similar to the ones applied for machines and operators' assignment will be performed in this step to identify the tools that should be assigned to various cells. This process is schematically shown in Figure 5.

The second section of the decoding structure is assigned to determining part assignment. This section is concerned with the need to perform 2, 2, 1, and 2 operations on four various parts. So, the first seven (i.e., $2+1+2+2$) elements of the row $c + 2$ need to be separated. This section is performed into four steps to assign each part to the required operations that should be performed on that part. To do so, separated numbers are sorted in ascending order and the first two positions are replaced with one (because two operations should be performed on part one). This process will be

	0.58	0.6	0.6	0.65	0.68	0.74	0.45
Part	2	2	1	1	1	1	1
Operation	1	1	1	1	1	1	1

Figure 6. Process of assigning parts to operations.

	0.15	0.19	0.41	0.75	0.82	0.79	0.32
Cell	1	1	1	2	2	1	1

Figure 7. Process of assigning parts and operation to various cells.

	0.49	0.8	0.14	0.42	0.92	0.79	0.96
Tool	3	5	1	3	6	5	6

Figure 8. Process of assigning tools to parts and cells.

pursued until all the parts are assigned to their required operations. This process is mainly shown in Figure 6.

The third section of the decoding structure is related to assigning parts and operations to different cells. The number of activities required for processing various parts on each cell equals seven. So, this section begins with separating the first seven elements of vector $c + 3$. Then, the integer number obtained from multiplying the number of cells by separated elements of vector $c + 3$ should be rounded up. The result of performing this procedure is shown in Figure 7.

The fourth section of the decoding structure is related to assigning tools to various parts and cells. This process starts with defining a set of N numbers. The ceil integer number resulting from multiplying N by numbers of vectors $c + 4$ is called A . Then, the A -th member of tools set is assigned to its pertinent part and cell. This process is shown in Figure 8.

The fifth section of the decoding structure is related to assigning operations to different machines located in two cells. To do so, a machine set with M members should be defined. Then, the ceil integer number obtained by multiplying the elements of $c + 5$ vector by M is called A . Then, the A -th member of the machine set should be assigned to its relevant operation. This process is shown in Figure 9.

The sixth section of the decoding structure is related to assigning operators to machines. To this end, an operator set with M' member should be defined in

	0.03	0.44	0.38	0.77	0.8	0.19	0.49
Machine	1	1	1	3	3	2	1

Figure 9. Process of assigning machines to operations.

	0.66	0.16	0.12	0.5	0.96	0.34	0.59
Operator	3	2	2	3	4	3	3

Figure 10. Procedure of assigning operators to machines.

each step of performing operator assignment. The ceil integer number obtained by multiplying the elements of vector $c + 6$ by a number of elements of the operator set is called A . Then, the A th element of the operator set should be assigned. This procedure will be pursued until all the operators are assigned to their pertinent machines. This process is shown in Figure 10.

4.2. MOVDO algorithm

The model proposed in this paper to formulate a DCMS is strictly NP-hard and exact algorithms cannot solve various test problems of the proposed model in a reasonable computational time. Therefore, an MOVDO algorithm is developed in this section to determine near-optimal solutions of the model's different test problems in a reasonable time. Also, three other solution algorithms (i.e., NSGA-II, MOPSO, and MOIWO) are considered to validate the performance of the proposed MOVDO algorithm in solving small-, medium-, and large-scale instances.

The MOVDO algorithm was firstly proposed by Mehdizadeh et al. [52] and developed by Hajipour et al. [53]. The main concept of this algorithm is inspired by a damping feature of mechanical vibrations. According to this algorithm, there is a logical relationship between hybrid optimization (finding the minimum value of an objective function subject to some realistic constraints) and the behavior of a fluctuating system in a damping position. When the energy source of an oscillator is interrupted, the oscillation range is reduced gradually and the oscillation process stops at the end. This process is called damping. The elastic properties of some special materials along with probability principles are embedded into the main structure of this algorithm to generate a new solution on each iteration and seek other neighborhood regions to find a better solution.

This algorithm is implemented into four various steps of decoding problem, defining fitness function, defining the mechanism of exploring neighborhood solution areas, and defining vibration damping procedure. A descending function is used to determine the reduction rate of oscillation rate into various iterations. According to this function, there is a reverse relationship between the oscillation domain and the possibility of generating a larger number of solutions. Also, this function is mainly embedded in the main structure

of the algorithm to reduce the oscillation range into various iterations. Therefore, the rate of generating solutions into various iterations should be reduced. The displacement relationship of the vibration theory can be used as the basis of computing vibration damping into a different iteration. Mathematical presentation damping vibration into different iterations is obtained as follows:

$$A_t = A_0 e^{\frac{-\gamma t}{2}}, \quad (35)$$

where A , A_0 , γ , and t are the oscillation range, initial oscillation range, damping coefficient, and a number of iterations of damping loop, respectively.

A damping coefficient is a parameter of controlling the reduction rate of the oscillation rate. The value assigned to this parameter is in inverse relation with the reduction rate of the oscillation rate. Therefore, choosing an appropriate value of this parameter to achieve appropriate solutions in a reasonable time is very significant. The other factor that has a great influence on both solutions' quality and computational time is the number of iterations that should be performed to reach the best possible solution. Increasing the number of iterations enables the algorithm to explore more feasible solution areas and obtain more solutions. Also, the main consequence of increasing the number of iterations increases the algorithm's convergence time. However, increasing the level of this parameter may not necessarily enhance the quality of the best solutions obtained through different iterations. Another factor that has an particular effect on the performance of the proposed MOVDO algorithm is the stopping criterion. Different conditions can be considered as factors in stopping the algorithm's implementation process. One of these factors is the maximum number of iterations. Another factor is to get zero value of the oscillation range. The best value of these factors should be calculated systematically to retain the algorithm's appropriate convergence and ensure its well performance to achieve proper solutions.

To explain the trend of the proposed algorithm, a pseudo code of MOVDO is exhibited, as shown in Figure 11. Figure 12 shows the pseudo-code of non-dominated sorting by the fuzzy concept.

4.3. Other meta-heuristic algorithms

Since there is no benchmark available to measure the performance of the proposed algorithms, three other meta-heuristic algorithms are developed with a similar solution structure to evaluate the performance of the proposed algorithm in solving the model's small-, medium-, and large-sized instances.

4.3.1. Non-dominated Sorting Genetic Algorithm-II (NSGA-II)

NSGA-II proposed by Deb et al. [54] is used to

```

Begin
  Input  $nPop$  (Population number,  $\gamma$  (damping coefficient); and  $\sigma$  (Rayleigh distribution constant)
  Initialize ( $X; A; L$  and  $t, t = 1$ );
  Evaluate solutions
  Perform non-dominate sorting by fuzzy concept according to fig. 12 and calculate ranks
  Calculate the crowding distance (CD)
  Sort population according to ranks and CDs
  For  $j = 1 : nPop$ 
     $P_j = population$ 
    For  $i = 1 : L$ 
       $Y = PERTURB(X)$ ; {Generate a new neighborhood solution}
       $\Delta = E(Y) - E(X)$ ;
      If  $\Delta \leq 0$  or  $(1 - e^{\frac{-\Delta^2}{2\sigma^2}} > Random(0,1))$ 
        Then,  $X = Y$ ; {Accept the movement if dominates final Pareto solution}
      End if
      Update ( $A$  and  $t$ ,  $A = A_0 e^{-\gamma t/2}$ ,  $t = t + 1$ )
    Until (Stop-Criterion)
  End for
   $Q_j = \text{new population}$ 
   $R_j = P_j \cup Q_j$ 
  Perform non-dominated sorting on  $R_j$  and calculate ranks
  Calculate the CD of  $R_j$ 
  Sort population according to ranks and CDs on  $R_j$ 
  Create  $P_{j+1}$  as size as population size (population =  $P_{j+1}$ )
End for
End

```

Figure 11. Pseudo code of MOVDO.

```

For  $k = 1 : n$  //calculate fuzzy J-dominance per solution in the population //
   $\mu(k) = 0$ 
  for  $i = 1 : n$ 
     $\mu(i) = 1$ 
    for  $j = 1 : m$  // compute fuzzy J-dominance per solution //
      if  $y_i(\bar{x}_i) - y_j(\bar{x}_k) < 0$ 
         $\mu_i^{dom}(\bar{x}_k <^F \bar{x}_i) = 0$ 
      elseif  $y_j(\bar{x}_i) - y_j(\bar{x}_k) < p_j$ 
         $\mu_i^{dom}(\bar{x}_k <^F \bar{x}_i) = \frac{y_j(\bar{x}_i) - y_j(\bar{x}_k)}{p_j}$ 
      else
         $\mu_i^{dom}(\bar{x}_k <^F \bar{x}_i) = 1$ 
      end
       $\mu^{dom}(\bar{x}_k <^F \bar{x}_i) = \mu^{dom}(\bar{x}_k <^F \bar{x}_i) \times \mu_i^{dom}(\bar{x}_k <^F \bar{x}_i)$ 
    End
  end
   $\mu(k) = \mu(k) + \mu^{dom}(\bar{x}_k <^F \bar{x}_i) - \mu(k) \times \mu^{dom}(\bar{x}_k <^F \bar{x}_i)$ 
End
End
End

```

Figure 12. Pseudo code of non-dominated sorting by the fuzzy concept.

evaluate the quality solutions obtained by the proposed MOVDO. This algorithm was developed based on the crowding distance and non-dominance techniques to determine better solutions and find the best possible Pareto front. This algorithm employs crossover and mutation operators to generate new solutions. Then,

combinations of offspring and previously generated solutions are conveyed to the next generation.

Non-dominance technique

The model proposed in this paper includes two conflicting objectives. We assume that X_1 and X_2 are

two different solutions obtained by NSGA-II. Solution X_1 can be dominated by solution X_2 if the following conditions are completely satisfied:

- The fitness value of solution X_1 should not be the worse fitness value of solution X_2 ;
- At least for one objective, the fitness value of solution X_1 is better than the corresponding fitness value of solution X_2 .

The solutions that are not dominated by other solutions are classified as the first Pareto front, and the solutions that are merely dominated by solutions of the first Pareto front are classified as the second Pareto. This process is pursued until all the solutions are classified on their appropriate Pareto front.

Crowding distance

This technique is mainly used to calculate the density of the solutions categorized on different Pareto fronts. This technique computes the distance of each solution from its neighboring solutions. The following formula can be used to compute the crowding distance value of each solution in a two-dimensional solution space:

$$CD_i = \sum_{j=1}^k \frac{f_{j,i+1}^p - f_{j,i-1}^p}{f_{j,\max}^p - f_{j,\min}^p}, \quad (36)$$

where the number of objective functions, value of the j -th objective of solutions $i+1$ and $i-1$ located on the p -th Pareto front, and the minimum and maximum values of the objective j obtained into Pareto p are denoted by k , $f_{j,i+1}^p$, $f_{j,i-1}^p$, $f_{j,\max}^p$, and $f_{j,\min}^p$, respectively.

Binary tournament selection operator

This operator is mainly used to select parent solutions of the crossover operator. According to this method, two members are randomly generated and members with a lower rank are chosen as a parent solution. If there is no difference between the ranks of the randomly selected solution, the one with a higher crowding distance value will be selected as a parent solution.

4.3.2. Multi-Objective Particle Swarm Optimization (MOPSO) algorithm

This algorithm was first developed to solve multi-objective constrained and unconstrained optimization problems by Coello et al. [55]. The procedure of this algorithm starts by generating random feasible solutions. Then, an update position mechanism is employed to improve the quality of solutions generated in various iterations. A repository archive is used to keep non-dominated solutions. Then, the solutions generated by the non-dominance technique are removed from the repository.

Update position

The main focus of the MOPSO algorithm is to use

the following equations to update the position and velocity of the particles into two-dimensional feasible solution space. This mechanism enables the algorithm to enhance the quality of solutions through various generations.

$$V_i^{t+1} = W \times V_i^t + C_1 \times r_1 \times (Pbest_i^t - X_i^t) + C_2 \times r_2 \times (gbest_i^t - X_i^t), \quad (37)$$

$$X_i^{t+1} = X_i^t + V_i^{t+1}, \quad (38)$$

where X_i^{t+1} , X_i^t , V_i^{t+1} , V_i^t , $Pbest_i^t$, $gbest_i^t$, C_1 , C_2 , r_1 , r_2 and W represent a personal position of particle i into iterations t and $t+1$, the velocity of particle i into iterations t and $t+1$, the best personal memory of particle i into iterations t , personal and global learning coefficients, random numbers generated between $[0, 1]$, and inertia weight, respectively.

Leader selection

The process of selecting a leader solution begins with classifying repository members into different grids. Then, a roulette wheel mechanism is used to elaborate on the possibility of choosing a grid with lower density. One of the members of the chosen grid is selected randomly to be known as a leader solution.

4.3.3. Multi-Objective Invasive Weeds Optimization (MOIWO) algorithm

MOIW algorithm is developed with the same solution structure to validate the performance of the proposed MOVDO algorithm in solving small- to large-scale problems. MOIWO is a population-based algorithm that works based on the behavior of invasive weeds. Similar to particles of MOPSO, invasive weeds of this algorithm are expanding and generating new weeds into the earth's surface by their pruning method. Firstly, they find appropriate farmlands to generate new weeds. According to this algorithm, the seeds that are scattered into special lands to be turned into invasive weeds. Consequently, a new set of invasive weeds will be generated around original weeds. The weeds with a greater growth into different segments of farming lands have more chances to survive. This issue leads to an increase in the fertility rate in the vicinity of these invasive weeds. Also, the main result of increasing the number of sprouts is a systematic reduction of both parent members and the distance between newly germinated invasive weeds.

In this paper, because of the good performance of MOIWO in Nabovati et al. [56], Keramatpour et al. [57], and Nabovati et al. [58], this algorithm was selected as an indicator for the efficiency of the MOVDO algorithm.

The main steps for implementing the MOIWO algorithm are as follows:

- Random generation of initial solutions and evaluation of their fitness values;
- Use of a fuzzy ranking method to prioritize population members;
- Permitting each member to generate several seeds with better population members. The seeds are generated based on the following equation:

$$seeds_i = floor \left(S_{\min} + (S_{\max} - S_{\min}) \cdot \left(\frac{np - rank_i}{np} \right) \right), \quad (39)$$

where $seeds_i$, S_{\min} , S_{\max} , np , $rank_i$ are the number of seeds generated by the i -th population member, minimum and maximum numbers of seeds generated by each seed, number of initial population, and grade of the i th member of the population, respectively.

- Increasing the breeding rate based on competition and update of standard deviation. The following equation is used to compute the standard deviation:

$$\sigma_{iter} = (iter_{\max} - iter)^n \cdot \frac{\sigma_{initial} - \sigma_{final}}{iter_{\max}^n} - \sigma_{final}, \quad (40)$$

where the index of the nonlinear formulation is denoted by n . The value of this parameter plays a crucial role in obtaining suitable solutions.

5. Performance metrics and parameter setting

5.1. Performance metrics

Six various metrics are considered in this section to evaluate the performance of the proposed algorithms. These metrics are listed as follows:

5.1.1. Mean Ideal Distance (MID)

This metric is mainly used to compute the distance of the Pareto solutions from an ideal solution. The solutions with a higher MID value will be better than the other solutions. The corresponding value of this metric is calculated as follows.

$$MID = \frac{\sum_{i=1}^n \sqrt{\left(\frac{f1_i - f1_{best}}{f1_{total}^{\max} - f1_{total}^{\min}} \right)^2 + \left(\frac{f2_i - f2_{best}}{f2_{total}^{\max} - f2_{total}^{\min}} \right)^2}}{n}, \quad (41)$$

where $f1_{total}^{\min}$, $f1_{total}^{\max}$, and n are the smallest and largest values of all the solutions obtained by the developed algorithm and the number of non-dominated solutions, respectively [59].

5.1.2. Spacing Metric (SM)

This metric is mainly used to compute the uniformity of

the solutions obtained by the algorithm. The solution with a lower spacing value will have a higher priority. The value of this metric is computed as follows [60]:

$$SM = \frac{\sum_{i=1}^{n-1} |\bar{d} - d_i|}{(n-1) \cdot \bar{d}}, \quad (42)$$

where d_i and \bar{d} are the Euclidean distance between iterative solutions of the non-dominated solutions obtained by the algorithm, respectively.

5.1.3. Diversification Metric (DM)

This metric is used to compute the extension of the obtained non-dominated algorithms [61]. The solution with a higher value of this metric will have better quality. The corresponding value of this metric is computed by:

$$DM = \sqrt{(f1_{total}^{\max} - f1_{total}^{\min})^2 + (f2_{total}^{\max} - f2_{total}^{\min})^2}, \quad (43)$$

where $f1_{total}^{\max}$, $f1_{total}^{\min}$, $f2_{total}^{\max}$, and $f2_{total}^{\min}$ are the maximum and minimum values of the first and second objectives, respectively.

5.1.4. Spread of Non-dominated Solutions (SNS)

This metric is mainly used to calculate the diversity of the obtained non-dominated solutions. The solutions with higher values of this metric will have better quality. The value of this metric is computed by [61]:

$$SNS = \sqrt{\frac{\sum_{i=1}^{n'} (MID - C_i)^2}{n' - 1}}, \quad (44)$$

where:

$$C_i = \sqrt{f1_i^2 + f2_i^2}.$$

5.1.5. Central Processing Unit (CPU) time

This metric is mainly used to compute the computational time required for solving various test problems.

5.2. Parameter setting

All the parameters used in this paper can be divided into two parts including the parameters of the model and algorithm. This section is devoted to tuning these parameters.

5.2.1. Structure of the model's parameters

Thirty different test problems are randomly generated to evaluate the performance of the developed algorithms that are categorized based on number of cells (C), number of parts (P), number of skill levels (S), number of operations (O), number of tools (L), number of workers (W), and number of machines (M). All the problems are solved three times and their average is used for evaluating the performance of the developed algorithms. The main features of these problems are shown in Table 2.

Table 2. General data on test problems.

Problem	<i>P</i>	<i>C</i>	<i>L</i>	<i>O</i>	<i>W</i>	<i>M</i>	<i>T</i>
1	4	2	3	2	2	3	10
2	4	2	3	2	2	3	10
3	4	2	3	2	3	3	10
4	6	2	3	2	2	3	10
5	6	2	3	2	2	3	10
6	8	2	3	2	2	3	10
7	8	2	3	3	2	3	10
8	10	2	3	3	3	4	10
9	10	2	3	3	3	4	10
10	12	2	3	3	3	4	10
11	12	2	3	3	3	4	10
12	15	3	3	4	3	4	15
13	15	3	3	4	3	4	15
14	20	3	3	4	4	5	15
15	20	3	3	4	4	5	15
16	25	3	3	4	4	5	15
17	25	3	3	4	4	5	15
18	30	3	3	4	4	5	15
19	30	3	3	4	4	5	15
20	40	3	3	4	5	6	15
21	40	3	3	4	5	6	15
22	45	4	3	5	5	6	20
23	45	4	3	5	5	6	20
24	50	4	3	5	5	6	20
25	50	4	3	5	5	6	20
26	55	4	3	5	6	7	20
27	60	4	3	6	6	7	20
28	60	4	3	6	6	7	20
29	70	4	3	6	6	7	20
30	70	4	3	6	6	7	20

5.2.2. Tuning algorithm parameters

The Taguchi method that is firstly proposed by Taguchi [62] is mainly used to calibrate the performance of the proposed meta-heuristic algorithms. This method can be used as an alternative of a full factorial experimental method to determine the best level of parameters in a reasonable computational time. The factors used in this method can be divided into two

groups: controllable and noise factors. The main purpose of this method is to determine the best controllable level so that the effect of the noise factor is minimized. To do so, a signal-to-noise ratio is used to investigate the performance of the developed algorithms and determine the best level of parameters. This ratio indicates the response variable's deviation and its corresponding value is calculated by:

$$(S/N) = -10 \log \left(\frac{S(Y^2)}{n} \right), \quad (45)$$

where Y and n are the response variables and a number of orthogonal arrays, respectively.

Since the model presented in this paper belongs to a class of multi-objective problems and the solutions obtained in each experiment are presented in the form of Pareto solutions, a response variable of each experiment is considered as a combination of the metrics used for evaluating algorithms' performance. Therefore, the response variable presented by Rahmati et al. [63], which encompasses two main features of the meta-heuristic algorithms (convergence and diversification), is used as a response variable of different experiments determined by the Taguchi method. The main advantage of using this response is to consider convergence and divergence features of the developed solution algorithms. Therefore, the following equation is employed to calculate the response values of different experiments:

$$MOCV = MID/DM, \quad (46)$$

where MID and DM are indicators of convergence and divergence features of solutions algorithms, respectively.

To use the Taguchi method, three different levels, as shown in Table 3, are defined for each factor. Then, Minitab software is used to calculate optimal values of the S/N ratio. According to this software, L^9 design is used for evaluating the performance of the NSGA-II and MODVO algorithms; meanwhile, L^{27} design is used for evaluating the performance of the MOIWO and MOPSO algorithms. Orthogonal arrays of designs and their corresponding response values are shown in Tables 4 to 7.

Each design is implemented three times and their mean is used as a response value. The corresponding value of the S/N ratio for each algorithm is shown in Figure 13.

6. Computational result

Experimental outputs of four developed algorithms on solving 30 different test problems in terms of five different metrics are shown in Table 8. Also, the average of the results in terms of previously defined

Table 3. Algorithm parameter ranges along with their levels.

Multi-objective algorithms	Parameters	Parameter level			Optimum value
		Level 1	Level 2	Level 3	
NSGA-II	Pc	0.7	0.8	0.9	0.9
	Pm	0.1	0.15	0.2	0.2
	N-pop	100	150	200	100
	Max iteration	$5 * N$	$10 * N$	$15 * N$	$15 * N$
MOPSO	C_1	1	1.4962	2	2
	C_2	1	1.4962	2	2
	W	0.6	0.7298	0.9	0.6
	N-Particle	100	150	200	100
	Max It	$5 * N$	$10 * N$	$15 * N$	$15 * N$
	N-Rep	50	70	100	70
	N-Grid	5	8	10	5
MOIWO	P-Max	100	150	200	100
	Initial sigma	0.3	0.4	0.5	0.5
	Max iteration	$5 * N$	$10 * N$	$15 * N$	$5 * N$
	N-Pop	100	150	200	200
	Final sigma	0.01	0.03	0.05	0.01
	S-Min	1	2	3	3
	S-Max	5	8	10	5
	N	2	3	4	4
	KF	1	2	3	1
	N-Archive	100	150	200	200
MOVDO	N-Pop	100	150	200	100
	Max It	$5 * N$	$10 * N$	$15 * N$	$15 * N$
	Gamma	0.005	0.05	0.5	0.005
	Sigma	1	1.5	2	1.5
	A	6	8	10	6

metrics for small-, medium-, and large-scale problems are shown in Table 9. The problems are classified into three groups of small-, medium-, and large-scale problems. According to this classification, Problems 1 – 10, 11 – 20, and 21 – 30 are considered small-,

medium-, and large-scale problems, respectively. All the problems are coded in MATLAB software. Also, they are solved using a PC with Core I2 CPU and 2 GB RAM. Moreover, the trends of all metrics over different test problems shown in Figure 14 reveal that MOVDO

Table 4. Computational results to tune MOIWO.

Run order	Max-It	N-Pop	P-Max	Initial sigma	Final sigma	S-Min	S-Max	n	KF	N-Archive	MOCV
1	1	1	1	1	1	1	1	1	1	1	0.107
2	1	1	1	1	2	2	2	2	2	2	0.1134
3	1	1	1	1	3	3	3	3	3	3	0.0589
4	1	2	2	2	1	1	1	2	2	2	0.1139
5	1	2	2	2	2	2	2	3	3	3	0.0943
6	1	2	2	2	3	3	3	1	1	1	0.0568
7	1	3	3	3	1	1	1	3	3	3	0.0695
8	1	3	3	3	2	2	2	1	1	1	0.0883
9	1	3	3	3	3	3	3	2	2	2	0.117
10	2	1	2	3	1	2	3	1	2	3	0.1175
11	2	1	2	3	2	3	1	2	3	1	0.061
12	2	1	2	3	3	1	2	3	1	2	0.1179
13	2	2	3	1	1	2	3	2	3	1	0.117
14	2	2	3	1	2	3	1	3	1	2	0.084
15	2	2	3	1	3	1	2	1	2	3	0.106
16	2	3	1	2	1	2	3	3	1	2	0.0599
17	2	3	1	2	2	3	1	1	2	3	0.0795
18	2	3	1	2	3	1	2	2	3	1	0.1141
19	3	1	3	2	1	3	2	1	3	2	0.1055
20	3	1	3	2	2	1	3	2	1	3	0.1172
21	3	1	3	2	3	2	1	3	2	1	0.0959
22	3	2	1	3	1	3	2	2	1	3	0.0525
23	3	2	1	3	2	1	3	3	2	1	0.1094
24	3	2	1	3	3	2	1	1	3	2	0.1154
25	3	3	2	1	1	3	2	3	2	1	0.0975
26	3	3	2	1	2	1	3	1	3	2	0.103
27	3	3	2	1	3	2	1	2	1	3	0.102

Table 5. Computational results to tune the NSGA-II.

Run order	P_{cr}	P_{mut}	Max-It	N-Pop	Response
1	1	1	1	1	0.069674
2	1	2	2	2	0.10049
3	1	3	3	3	0.056383
4	2	1	2	3	0.06884
5	2	2	3	1	0.094097
6	2	3	1	2	0.063253
7	3	1	3	2	0.078815
8	3	2	1	3	0.07052
9	3	3	2	1	0.063952

has better performance in terms of small-, medium-, and large-scale MID, DM, and CPU time metrics. Also, there is no meaningful difference between the performances of the developed algorithms in terms of SM and SNS metrics.

6.1. Ranking algorithms

A Multiple Attribute Decision Making (MADM) method is developed in this section to evaluate the overall performance of the developed algorithm and determine the algorithm with a higher performance in solving small-, medium-, and large-scale problems. To do so, an AHP-PROMETHEE method is used in this section to determine the overall priority of algorithms

Table 6. Computational results to tune MOPSO.

Run order	C_1	C_2	W	Max-It	N -Pop	N -Rep	N -Grid	Response
1	1	1	1	1	1	1	1	0.054988
2	1	1	1	1	2	2	2	0.073798
3	1	1	1	1	3	3	3	0.065228
4	1	2	2	2	1	1	1	0.085455
5	1	2	2	2	2	2	2	0.066644
6	1	2	2	2	3	3	3	0.067903
7	1	3	3	3	1	1	1	0.07864
8	1	3	3	3	2	2	2	0.067293
9	1	3	3	3	3	3	3	0.068945
10	2	1	2	3	1	2	3	0.101934
11	2	1	2	3	2	3	1	0.08617
12	2	1	2	3	3	1	2	0.068239
13	2	2	3	1	1	2	3	0.080162
14	2	2	3	1	2	3	1	0.063967
15	2	2	3	1	3	1	2	0.080303
16	2	3	1	2	1	2	3	0.059683
17	2	3	1	2	2	3	1	0.071043
18	2	3	1	2	3	1	2	0.083273
19	3	1	3	2	1	3	2	0.058213
20	3	1	3	2	2	1	3	0.069766
21	3	1	3	2	3	2	1	0.059577
22	3	2	1	3	1	3	2	0.072366
23	3	2	1	3	2	1	3	0.070046
24	3	2	1	3	3	2	1	0.061728
25	3	3	2	1	1	3	2	0.083507
26	3	3	2	1	2	1	3	0.059169
27	3	3	2	1	3	2	1	0.057342

in terms of previously defined metrics. According to this method, metrics and algorithms are considered as criteria and alternatives. This procedure is performed at two different phases: determining criteria weights and ranking alternatives to identify the algorithm with better performance in solving various test problems. The computational procedure of the Analytical Hierarchy Process (AHP) method starts by aggregating the opinions of three different experts to define the initial decision matrix. Then, the following formula is used to determine the relative weights of metrics. The results

are shown in Table 10.

$$\omega_i = \frac{\sum_{j=1}^n f_{ij}}{n}. \quad (47)$$

The preference ranking organization method for enrichment evaluation technique (PROMETHEE) is used to rank alternatives based on the weights of criteria obtained by AHP. This method employs preference and indifference words to determine the overall performance of algorithms. Also, a pairwise comparison matrix of

Table 7. Computational results to tune MOVDO.

	A	Sigma	Gamma	Max-It	N-Pop	MOCV
Run 1	1	1	1	1	1	0.096598
Run 2	1	1	1	1	2	0.10548
Run 3	1	1	1	1	3	0.060166
Run 4	1	2	2	2	1	0.108211
Run 5	1	2	2	2	2	0.088769
Run 6	1	2	2	2	3	0.059021
Run 7	1	3	3	3	1	0.071328
Run 8	1	3	3	3	2	0.084099
Run 9	1	3	3	3	3	0.107389
Run 10	2	1	2	3	1	0.114387
Run 11	2	1	2	3	2	0.066034
Run 12	2	1	2	3	3	0.107968
Run 13	2	2	3	1	1	0.109632
Run 14	2	2	3	1	2	0.079993
Run 15	2	2	3	1	3	0.100861
Run 16	2	3	1	2	1	0.059857
Run 17	2	3	1	2	2	0.077809
Run 18	2	3	1	2	3	0.107935
Run 19	3	1	3	2	1	0.096043
Run 20	3	1	3	2	2	0.107713
Run 21	3	1	3	2	3	0.088635
Run 22	3	2	1	3	1	0.056473
Run 23	3	2	1	3	2	0.101529
Run 24	3	2	1	3	3	0.104666
Run 25	3	3	2	1	1	0.094701
Run 26	3	3	2	1	2	0.094234
Run 27	3	3	2	1	3	0.093068

alternatives with respect to each criterion is performed based on a predefined superiority function in the range of $[0 \ 1]$ to identify the domination of alternatives. The following formula is used to define the dominance of two different alternatives (a and b) with respect each other:

$$P_j(a, b) = P[d_j(a, b)]. \quad (48)$$

The main steps of PROMETHEE to determine the overall performance of the developed algorithms are as follows:

- Determine the difference of alternatives with respect to each metric. The following formula is used to

compute the difference between alternatives:

$$d_j(a, b) = f_j(a) - f_j(b). \quad (49)$$

This term is useful for both positive and negative metrics in which the following conditions are satisfied:

$f_j(a) > f_j(b)$: For positive metrics,

$f_j(a) < f_j(b)$: For negative metrics.

- Compute the superiority function of alternatives. Eq. (48) is used to compute the corresponding values of this function.

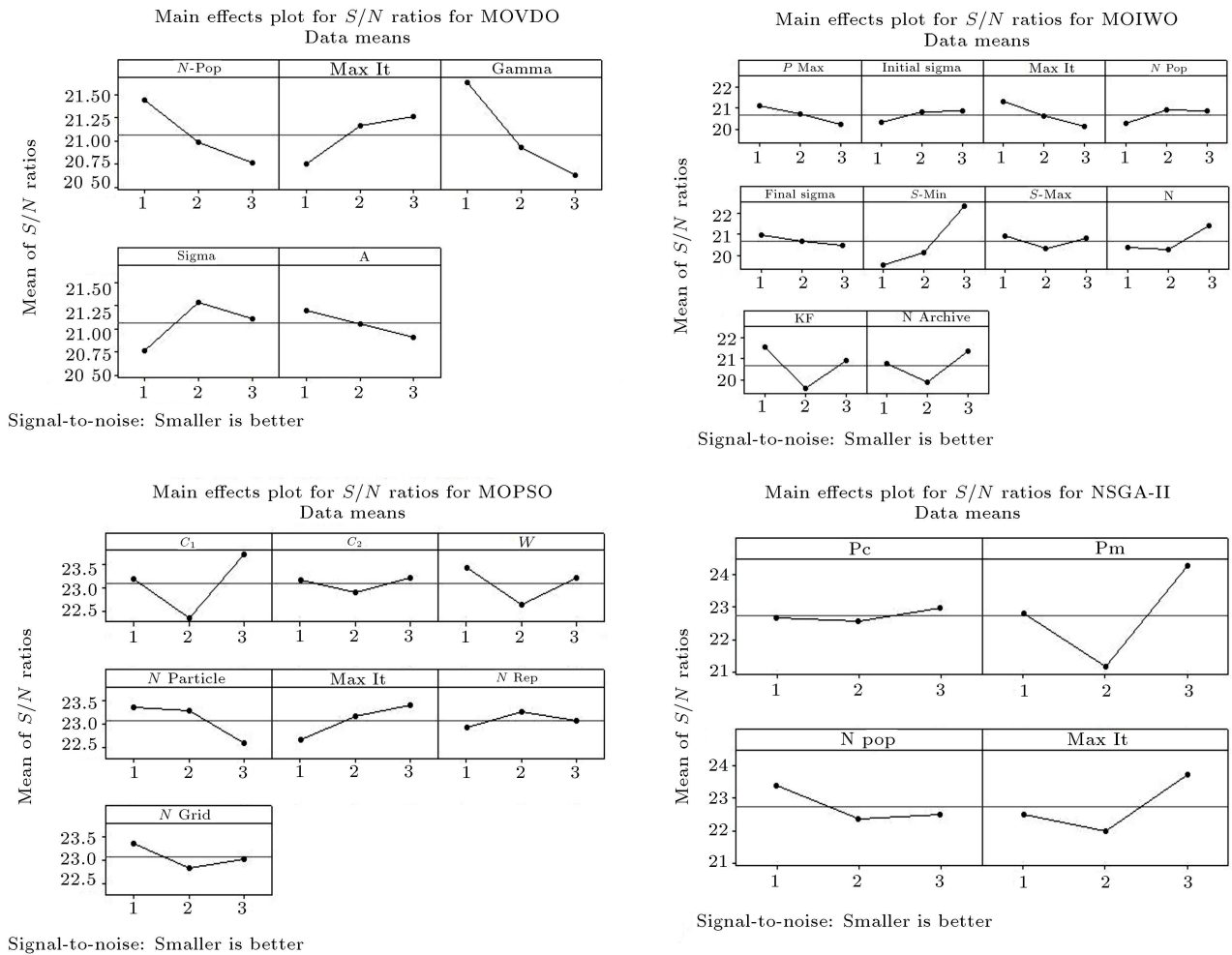


Figure 13. S/N ratio plots of the proposed algorithms.

- Compute the weighted sum of the superiority function. The following equation is used to calculate the corresponding weighted sum values of superiority functions for each alternative:

$$\pi(a, b) = \sum_{j=1}^k P_j(a, b) w_j, \quad (50)$$

$$\pi(b, a) = \sum_{j=1}^k P_j(b, a) w_j. \quad (51)$$

- Compute the positive output stream that is mainly used to compute the dominance rate of each alternative compared to other alternatives. The algorithm with a higher value of this term will have greater priority. The following formula can be used to compute this term:

$$\phi^+(a) = \frac{1}{n-1} \sum_{x \in A} \pi(a, x). \quad (52)$$

- Compute the negative input stream. This term is mainly used to compute the superiority of other

algorithms over each algorithm. The algorithm with a lower value of this term will have a higher rank. The following formula can be used to compute this term:

$$\phi^-(a) = \frac{1}{n-1} \sum_{x \in A} \pi(x, a). \quad (53)$$

- Compute pure stream. This term is mainly used to compute the overall priority of alternatives and select the algorithm with higher performance in solving small-, medium-, and large-scale problems. Using this term, the alternative with a higher value of this term will be selected as the best algorithm. The following formula is used to compute the corresponding value of this term for all alternatives and select the best one.

$$\phi(a) = \phi^+(a) - \phi^-(a). \quad (54)$$

The result of implementing this method on decision lab software is shown in Tables 11 to 13. The results show that the proposed MOVD0 algorithm has a

Table 8. Experimental outputs of the four algorithms.

Problem no.	NSGA-II		MOPSO		MOIWO		MOVDO		Epsilon constraint	
	MID	DM	MID	DM	MID	DM	MID	DM	MID	DM
1	7465	1717	6249	1731	6645	5767	5352	6033	4963	6324
2	9913	1196	5827	1559	5597	4273	5223	4579	5121	4734
3	14564	941	6698	1733	5793	3729	6255	4144	5989	4326
4	13739	2303	7886	3298	8864	5684	7636	6912	7432	7048
5	10871	2419	7926	3497	10687	7057	7916	8152	NA	NA
6	19984	4964	11497	6815	14170	8096	11625	11411	NA	NA
7	21487	2488	8460	5898	11039	6361	9884	8773	NA	NA
8	19277	11706	15020	12833	20296	9552	14808	17663	NA	NA
9	22196	5539	13183	8481	17103	8104	13556	12408	NA	NA
10	37766	6732	23012	7744	17214	7880	18271	13193	NA	NA
11	28346	3915	26075	8452	27471	10291	17787	13546	NA	NA
12	37418	7294	29301	9825	26240	12052	23192	12561	NA	NA
13	42791	5648	34439	11433	32463	15167	25732	14599	NA	NA
14	42954	7077	35167	12446	34403	15444	25000	17759	NA	NA
15	132823	9055	52895	11897	30122	13061	30051	19085	NA	NA
16	48165	12640	40868	15306	43593	19393	25169	16539	NA	NA
17	98524	32998	61609	21040	53810	16581	39997	33013	NA	NA
18	90276	22102	53644	17491	46007	16721	33559	25285	NA	NA
19	82186	14695	65641	25333	62694	26905	47617	37003	NA	NA
20	99153	19250	72104	22182	65561	22593	52168	26215	NA	NA
21	188951	24971	109700	28009	87827	25198	72899	27780	NA	NA
22	142737	18937	103019	27639	94712	25775	73693	40921	NA	NA
23	141422	15764	115137	30184	112730	27655	84853	38345	NA	NA
24	153682	14666	123346	29297	116905	30575	92061	42174	NA	NA
25	165284	11982	137614	29682	130089	34434	104849	43615	NA	NA
26	182542	19126	142904	33022	138162	34411	103916	49791	NA	NA
27	205600	24928	155329	36554	152827	36318	109377	39088	NA	NA
28	161726	18356	124264	28990	121245	29471	88872	35475	NA	NA
29	129380	14685	99411	23192	96996	23577	71098	31460	NA	NA
30	226317	15486	133090	31137	162490	24731	137523	44645	NA	NA

Problem no.	NSGA-II		MOPSO		MOIWO		MOVDO		Epsilon constraint	
	SM	SNS	SM	SNS	SM	SNS	SM	SNS	SM	SNS
1	270	2064	255	3053	1350	4921	805	8395	954	8832
2	203	3460	268	2958	969	3587	606	9555	1016	1104
3	182	5626	339	3520	804	3049	540	12837	897	15859
4	497	4132	602	2677	1132	4806	846	10805	1256	12865
5	700	1934	651	1117	1442	6232	1057	6912	NA	NA
6	979	5404	1226	2973	1373	6731	1250	13820	NA	NA
7	468	7028	1202	1428	1025	5179	967	13716	NA	NA
8	2069	2897	2110	3151	1087	7656	1591	11009	NA	NA
9	1003	5224	1432	2140	1243	6791	1252	12758	NA	NA
10	1548	7798	1311	4968	1340	7869	1104	6969	NA	NA

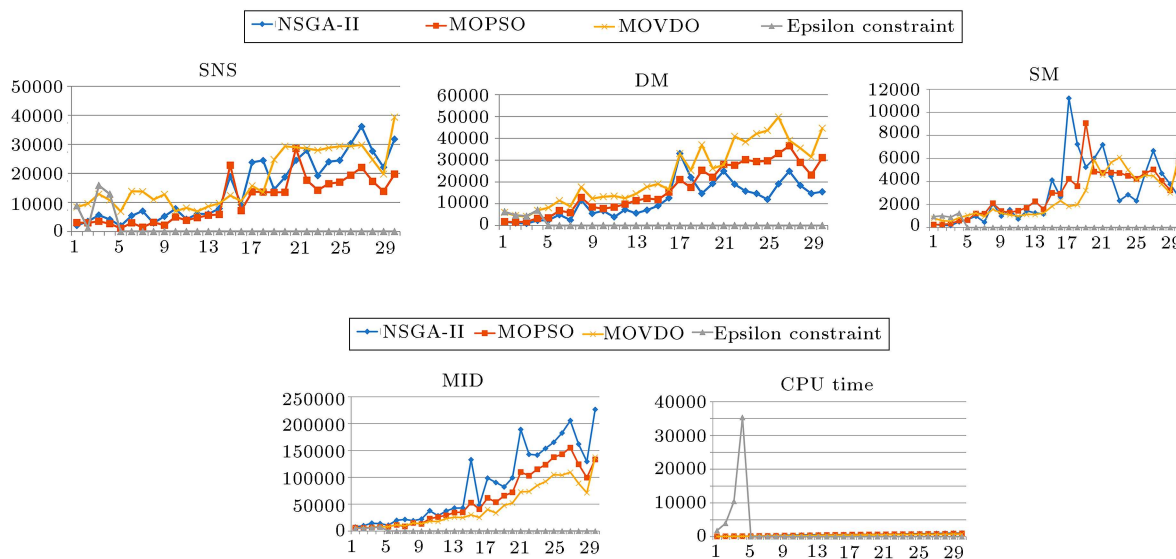
Table 8. Experimental outputs of the four algorithms (continued).

Problem no.	NSGA-II		MOPSO		MOIWO		MOVDO		Epsilon constraint	
	SM	SNS	SM	SNS	SM	SNS	SM	SNS	SM	SNS
11	741	4203	1455	3848	2020	9525	989	8109	NA	NA
12	1452	6078	1723	4795	2181	11484	1141	6900	NA	NA
13	1274	6005	2278	5575	3372	15902	1126	8634	NA	NA
14	1174	7895	1584	5857	1944	11975	1256	9684	NA	NA
15	4126	18855	3012	22825	3271	12041	1837	12381	NA	NA
16	2659	9259	2938	7155	3424	16514	2315	10457	NA	NA
17	11223	23784	4227	13668	2864	17306	1836	15714	NA	NA
18	7235	24441	3597	13546	3113	16016	1980	13470	NA	NA
19	5247	14433	9071	13458	14109	36235	3225	24735	NA	NA
20	6003	18697	4857	13500	2396	17160	5875	29315	NA	NA
21	7181	24491	4753	28551	3475	22715	4549	28966	NA	NA
22	4463	27902	4772	17618	3519	22492	5647	28557	NA	NA
23	2334	19209	4739	14211	4102	25046	6083	27973	NA	NA
24	2856	24017	4491	16430	4727	26571	5004	28792	NA	NA
25	2313	24479	4228	16946	5644	29373	4143	29319	NA	NA
26	4626	30202	4700	19359	5032	30069	4531	29424	NA	NA
27	6667	36156	5041	22030	4879	32166	4488	29793	NA	NA
28	4705	27649	4059	17246	4130	25931	3758	24674	NA	NA
29	3764	22119	3247	13797	3304	20745	3006	19739	NA	NA
30	2672	31814	5667	19734	6817	18304	6175	39369	NA	NA

Problem no.	NSGA-II	MOPSO	MOIWO	MOVDO	Epsilon constraint
	CPU time (sec)	CPU time (sec)	CPU time (sec)	CPU time (sec)	CPU time (sec)
1	80	73	67	59	1807
2	116	114	83	83	3954
3	139	137	94	99	10463
4	143	148	142	115	35329
5	160	163	147	125	NA
6	170	205	195	152	NA
7	252	228	219	186	NA
8	338	263	229	221	NA
9	361	312	233	241	NA
10	409	313	298	272	NA
11	456	351	334	304	NA
12	521	384	374	341	NA
13	538	413	411	363	NA
14	539	521	467	407	NA
15	556	529	474	415	NA
16	605	548	516	445	NA
17	616	610	524	466	NA
18	642	612	606	496	NA
19	664	648	620	515	NA
20	673	659	652	529	NA
21	696	686	686	551	NA
22	740	729	710	581	NA
23	777	734	721	595	NA
24	838	736	724	613	NA
25	871	767	750	637	NA
26	886	777	768	648	NA
27	919	806	768	665	NA
28	1010	833	769	696	NA
29	1032	953	774	735	NA
30	1071	962	768	747	NA

Table 9. Average experimental outputs of four algorithms.

Problems	Algorithm	SM	MID	DM	SNS	CPU time
Average 1–10	NSGA-II	792	17726	4000	4557	217
Average 11–20		4113	70264	13467	13365	581
Average 21–30		4158	169764	17890	26804	884
Average 1–10	MOPSO	940	10576	5359	2798	195
Average 11–20		3474	47174	15540	10423	527
Average 21–30		4570	124381	29771	18592	798
Average 1–10	MOIWO	1177	11741	6650	5682	170
Average 11–20		3869	42236	16821	16416	497
Average 21–30		4563	121398	29214	25341	744
Average 1–10	MOVDO	1002	10053	9327	10678	155
Average 11–20		2158	32027	21561	13940	428
Average 21–30		4739	93914	39329	28660	647
Average 1–4	Epsilon constraint	1031	5876	5608	9665	12888

**Figure 14.** Graphs of the metrics over test problems.**Table 10.** Pairwise comparison matrix of evaluating metrics.

Metrics	MID	DM	CPU time (sec)	SM	SNS	Weight
MID	1	2	1.5	3	4	0.358
DM	0.5	1	1	2	2	0.202
CPU time	0.67	1	1	2	3	0.232
SM	0.33	0.67	0.5	1	2	0.123
SNS	0.25	0.5	0.33	0.67	1	0.085

Table 11. Results of ranking alternatives for small-sized problems.

	Decision matrix					ϕ	Rank
	SM	MID	DM	SNS	CPU time		
NSGA-II	792	17726	4000	4557	217	−0.70	4
MOPSO	940	10576	5359	2798	195	−0.07	2
MOIWO	1117	11741	6650	5682	170	−0.07	3
MOVDO	1002	10053	9327	10678	155	0.84	1
Epsilon constraint	1031	5876	5608	9665	12888	−0.85	5

Table 12. Results of ranking alternatives for medium-sized problems.

	Decision matrix					ϕ	Rank
	SM	MID	DM	SNS	CPU time		
NSGA-II	4113	70264	13467	13365	581	−0.94	4
MOPSO	3474	47174	15540	10423	527	−0.31	3
MOIWO	3869	42236	16821	16416	497	0.31	2
MOVDO	2158	32027	21561	13940	428	0.94	1

Table 13. Results of ranking alternatives for large-sized problems.

	Decision matrix					ϕ	Rank
	SM	MID	DM	SNS	CPU time		
NSGA-II	4158	169764	17890	26804	884	−0.64	4
MOPSO	4570	124381	29771	18592	798	−0.26	3
MOIWO	4563	121398	29214	25341	744	0.14	2
MOVDO	4739	93914	39329	28660	647	0.75	1

better performance in solving small-, medium-, and large-scale problems.

7. Conclusion

In this paper, a novel bi-objective dynamic Cell Formation Problem (CFP) was proposed under a dynamic environment. The model aimed to minimize total costs and maximize total operators' efficiency. Since the model was strictly NP-hard and exact algorithms could not find global optimum solutions to medium- and large-scale problems in a reasonable computational time, a Multi-Objective Vibration Damping Optimization (MOVDO) algorithm was proposed with a new solution structure to solve different test problems. Furthermore, since there is no benchmark available in the literature to validate the performance of the developed meta-heuristic algorithm, three other algorithms were developed with the same solution structure to validate the performance of the developed algorithm. The Taguchi method was used to calibrate the main parameters of the developed algorithms and enhance their performance in solving the models in various instances. Statistical tests implemented to compare

the performance of the developed algorithms revealed that the proposed MOVDO algorithm had a better performance in terms of Mean Ideal Distance (MID), Diversification Metric (DM), and Central Processing Unit (CPU) time metrics. Also, there was a minor meaningful difference among the performances of the developed algorithms in terms of Spread of Non-dominated Solutions (SNS) and Spacing Metrics (SM). Finally, an AHP-PROMETHEE approach was performed to evaluate the overall performance of the developed algorithms in solving small-, medium-, and large-scale problems. The results demonstrated that the proposed MOVDO algorithm had a better performance in solving various small-, medium-, and large-scale problems.

References

1. Sofianopoulou, S. "Manufacturing cells design with alternative process plans and/or replicate machines", *International Journal of Production Research*, **37**(3), pp. 707–720 (1999).
2. Guerrero, F., Lozano, S., Smith, K.A., et al. "Manufacturing cell formation using a new self-organizing

- neural network”, *Computers and Industrial Engineering*, **42**(2), pp. 377–382 (2002).
3. Soleymanpour, M., Vrat, P., and Shankar, R. “A transiently chaotic neural network approach to the design of cellular manufacturing”, *International Journal of Production Research*, **40**(10), pp. 2225–2244 (2002).
 4. Logendran, R. and Karim, Y. “Design of manufacturing cells in the presence of alternative cell locations and material transporters”, *Journal of the Operational Research Society*, **54**(10), pp. 1059–1075 (2003).
 5. Spiliopoulos, K. and Sofianopoulou, S. “Designing manufacturing cells: A staged approach and a tabu search algorithm”, *International Journal of Production Research*, **41**(11), pp. 2531–2546 (2003).
 6. Prabhakaran, G., Asokan, P., Girish, B.S., et al. “Machine cell formation for cellular manufacturing systems using an ant colony system approach”, *The International Journal of Advanced Manufacturing Technology*, **25**(9), pp. 1013–1019 (2005).
 7. Defersha, F.M. and Chen, M. “A comprehensive mathematical model for the design of cellular manufacturing systems”, *International Journal of Production Economics*, **103**(2), pp. 767–783 (2006).
 8. Saidi-Mehrabad, M. and Safaei, N. “A new model of dynamic cell formation by a neural approach”, *The International Journal of Advanced Manufacturing Technology*, **33**(9), pp. 1001–1009 (2007).
 9. Defersha, F.M. and Chen, M. “A linear programming embedded genetic algorithm for an integrated cell formation and lot sizing considering product quality”, *European Journal of Operational Research*, **187**(1), pp. 46–69 (2008).
 10. Tavakkoli-Moghaddam, R., Safaei, N., and Sassani, F. “A new solution for a dynamic cell formation problem with alternative routing and machine costs using simulated annealing”, *Journal of the Operational Research Society*, **59**(4), pp. 443–454 (2008).
 11. Mahdavi, I., Teymourian, E., Tahami Baher, N., et al. “An integrated model for solving cell formation and cell layout problem simultaneously considering new situations”, *Journal of Manufacturing Systems*, **32**(4), pp. 655–663 (2013).
 12. Majazi Dalfard, V. “New mathematical model for problem of dynamic cell formation based on number and average length of intra and intercellular movements”, *Applied Mathematical Modelling*, **37**(4), pp. 1884–1896 (2013).
 13. Paydar, M.M. and Saidi-Mehrabad, M. “A hybrid genetic-variable neighborhood search algorithm for the cell formation problem based on grouping efficacy”, *Computers and Operations Research*, **40**(4), pp. 980–990 (2013).
 14. Salarian, R., Fazlollahtabar, H., and Mahdavi, I. “Inter-cell movement minimisation in a cellular manufacturing system having stochastic parameters”, *International Journal of Services and Operations Management*, **17**(1), pp. 67–87 (2014).
 15. Bychkov, I. and Batsyn, M. “An efficient exact model for the cell formation problem with a variable number of production cells”, *Computers & Operations Research*, **91**, pp. 112–120 (2018).
 16. Zohrevand, A.M., Rafiei, H., and Zohrevand, A.H. “Multi-objective dynamic cell formation problem: A stochastic programming approach”, *Computers & Industrial Engineering*, **98**, pp. 323–332 (2016).
 17. Mahdavi, I., Aalaei, A., Paydar, M.M., et al. “A new mathematical model for integrating all incidence matrices in multi-dimensional cellular manufacturing system”, *Journal of Manufacturing Systems*, **31**(2), pp. 214–223 (2012).
 18. Bagheri, M. and Bashiri, M. “A new mathematical model towards the integration of cell formation with operator assignment and inter-cell layout problems in a dynamic environment”, *Applied Mathematical Modelling*, **38**(4), pp. 1237–1254 (2014).
 19. Saidi-Mehrabad, M., Paydar, M.M., and Aalaei, A. “Production planning and worker training in dynamic manufacturing systems”, *Journal of Manufacturing Systems*, **32**(2), pp. 308–314 (2013).
 20. Paydar, M.M., Saidi-Mehrabad, M., and Kia, R. “Designing a new integrated model for dynamic cellular manufacturing systems with production planning and intra-cell layout”, *International Journal of Applied Decision Sciences*, **6**(2), pp. 117–143 (2013).
 21. Mehdizadeh, E. and Rahimi, V. “An integrated mathematical model for solving dynamic cell formation problem considering operator assignment and inter/intra cell layouts”, *Applied Soft Computing*, **42**, pp. 325–341 (2016).
 22. Mehdizadeh, E., Niaki, S.V.D., and Rahimi, V. “A vibration damping optimization algorithm for solving a new multi-objective dynamic cell formation problem with workers training”, *Computers & Industrial Engineering*, **101**, pp. 35–52 (2016).
 23. Feng, H., Da, W., Xi, L., et al. “Solving the integrated cell formation and worker assignment problem using particle swarm optimization and linear programming”, *Computers & Industrial Engineering*, **110**, pp. 126–137 (2017).
 24. Forghani, K. and Fatemi Ghomi, S.M.T. “A queuing theory-based approach to designing cellular manufacturing systems”, *Scientia Iranica*, **26**(3), pp. 1865–1880 (2019).
 25. Hashemoghli, A., Mahdavi, I., and Tajdin, A. “A novel robust possibilistic cellular manufacturing model considering worker skill and product quality”, *Scientia Iranica*, **26**(1), pp. 538–556 (2019).
 26. Vakharia, A.J. and Chang, Y.L. “Cell formation in group technology: A combinatorial search approach”, *International Journal of Production Research*, **35**(7), pp. 2025–2044 (1997).
 27. Ravichandran, K.S. and Chandra Sekhara Rao, K. “A new approach to fuzzy part-family formation in cellular manufacturing systems”, *The International Journal of*

Advanced Manufacturing Technology, **18**(8), pp. 591–597 (2001).

28. Lozano, S., Canca, D., Guerrero, F., et al. "Machine grouping using sequence-based similarity coefficients and neural networks", *Robotics and Computer-Integrated Manufacturing*, **17**(5), pp. 399–404 (2001).
29. Lozano, S., Dobado, D., Larrañeta, J., et al. "Modified fuzzy C-means algorithm for cellular manufacturing", *Fuzzy Sets and Systems*, **126**(1), pp. 23–32 (2002).
30. Wu, T.-H., Low, C., and Wu, W.T. "A tabu search approach to the cell formation problem", *The International Journal of Advanced Manufacturing Technology*, **23**(11), pp. 916–924 (2004).
31. Diaby, M. and Nsakanda, A.L. "Large-scale capacitated part-routing in the presence of process and routing flexibilities and setup costs", *Journal of the Operational Research Society*, **57**(9), pp. 1100–1112 (2006).
32. Lei, D. and Wu, Z. "Tabu search for multiple-criteria manufacturing cell design", *The International Journal of Advanced Manufacturing Technology*, **28**(9), pp. 950–956 (2006).
33. Wu, X., Chu, C.-H., Wang, Y., et al. "A genetic algorithm for cellular manufacturing design and layout", *European Journal of Operational Research*, **181**(1), pp. 156–167 (2007).
34. Mukattash, A.M., Adil, M.B., and Tahboub, K.K. "Heuristic approaches for part assignment in cell formation", *Computers & Industrial Engineering*, **42**(2), pp. 329–341 (2002).
35. James, T.L., Brown, E.C., and Keeling, K.B. "A hybrid grouping genetic algorithm for the cell formation problem", *Computers & Operations Research*, **34**(7), pp. 2059–2079 (2007).
36. Ghotboddini, M.M., Rabbani, M., and Rahimian, H. "A comprehensive dynamic cell formation design: Benders' decomposition approach", *Expert Systems with Applications*, **38**(3), pp. 2478–2488 (2011).
37. Martins, I.C., Pinheiro, R.G.S., Protti, F., et al. "A hybrid iterated local search and variable neighborhood descent heuristic applied to the cell formation problem", *Expert Systems with Applications*, **42**(22), pp. 8947–8955 (2015).
38. Kao, Y. and Li, Y.L. "Ant colony recognition systems for part clustering problems", *International Journal of Production Research*, **46**(15), pp. 4237–4258 (2008).
39. Rafiei, H. and Ghodsi, R. "A bi-objective mathematical model toward dynamic cell formation considering labor utilization", *Applied Mathematical Modelling*, **37**(4), pp. 2308–2316 (2013).
40. Zeidi, J.R., Javadian, N., Tavakkoli-Moghaddam, R., et al. "A hybrid multi-objective approach based on the genetic algorithm and neural network to design an incremental cellular manufacturing system", *Computers & Industrial Engineering*, **66**(4), pp. 1004–1014 (2013).
41. Shiyas, C. R. and Madhusudanan Pillai, V. "A mathematical programming model for manufacturing cell formation to develop multiple configurations", *Journal of Manufacturing Systems*, **33**(1), pp. 149–158 (2014).
42. Rezazadeh, H., Mahini, R. and Zarei, M. "Solving a dynamic virtual cell formation problem by linear programming embedded particle swarm optimization algorithm", *Applied Soft Computing*, **11**(3), pp. 3160–3169 (2011).
43. Buruk Sahin, Y. and Alpay, S. "A metaheuristic approach for a cubic cell formation problem", *Expert Systems with Applications*, **65**, pp. 40–51 (2016).
44. Durga Rajesh, K.V., Mani Krishna, M., Abid Ali, M.D., et al. "A modified hybrid similarity coefficient based method for solving the cell formation problem in cellular manufacturing system", *Materials Today: Proceedings*, **4**(2, Part A), pp. 1469–1477 (2017).
45. Behnia, B., Mahdavi, I., Shirazi, B., et al. "A bi-level bi-objective mathematical model for cellular manufacturing system applying evolutionary algorithms", *Scientia Iranica*, **26**(4), pp. 2541–2560 (2019).
46. Rostami, A., Paydar, M.M., and Asadi-Gangraj, E. "Dynamic virtual cell formation considering new product development", *Scientia Iranica*, **27**(4), pp. 2093–2107 (2020).
47. Tavanayi, M., Hafezalkotob, A., and Valizadeh, J. "Cooperative cellular manufacturing system: A cooperative game theory approach", *Scientia Iranica*, **28**(5), pp. 2769–2788 (2021).
48. Hajipour, V., Zanjirani Farahani, R., and Fattahi, P. "Bi-objective vibration damping optimization for congested location-pricing problem", *Computers & Operations Research*, **70**, pp. 87–100 (2017).
49. Tavakkoli-Moghaddam, R., Noshafagh, S.V., and Taleizadeh, A.A. "Pricing and location decisions in multi-objective facility location problem with M/M/m/k queuing systems", *Engineering Optimization*, **49**(1), pp. 136–160 (2017).
50. Hajipour, V., Khodakarami, V., and Tavana, M. "The redundancy queuing location-allocation problem: A novel approach", *IEEE Transactions on Engineering Management*, **61**(3), pp. 534–544 (2014).
51. Hajipour, V., Fattahi, P., Tavana, M., et al. "Multi-objective multi-layer congested facility location-allocation problem optimization with Pareto-based meta-heuristics", *Applied Mathematical Modeling*, **40**, pp. 4948–4969 (2016).
52. Mehdizadeh, E., Tavakkoli-Moghaddam, R., and Yazdani, M. "A vibration damping optimization algorithm for a parallel machines scheduling problem with sequence-independent family setup times", *Applied Mathematical Modelling*, **39**(22), pp. 6845–6859 (2015).
53. Hajipour, V., Mehdizadeh, E., and Tavakkoli-Moghaddam, R. "A novel Pareto-based multi-objective vibration damping optimization algorithm to solve

- multi-objective optimization problems”, *Scientia Iranica*, **21**(6), pp. 2368–2378 (2014).
54. Deb, K., Pratap, A., Agarwal, S., et al. “A fast and elitist multiobjective genetic algorithm: NSGA-II”, *IEEE Transactions on Evolutionary Computation*, **6**(2), pp. 182–197 (2002).
 55. Coello, C.C., Lamont, G.B., Veldhuizen, V., et al., *Evolutionary Algorithms for Solving Multi-Objective Problems*, Springer Science & Business Media (2007).
 56. Nabovati, H., Haleh, H., and Vahdani, B. “Fuzzy multi-objective optimization algorithms for solving multi-mode automated guided vehicles by considering machine break time and artificial neural network”, *Neural Network World: International Journal on Neural and Mass-Parallel Computing and Information Systems*, **28**(3), pp. 255–283 (2018).
 57. Keramatpour, M., AkhavanNiaki, S.T., and Pasandideh, S.H.R. “A bi-objective two-level newsvendor problem with discount policies and budget constraint”, *Computers & Industrial Engineering*, **120**, pp. 192–205 (2018).
 58. Nabovati, H., Haleh, H., and Vahdani, B. “Multi-objective invasive weeds optimisation algorithm for solving simultaneous scheduling of machines and multi-mode automated guided vehicles”, *European Journal of Industrial Engineering*, **14**(2), pp. 165–188 (2020).
 59. Kundu, D., Suresh, K., Ghosh, S., et al. “Multi-objective optimization with artificial weed colonies”, *Information Sciences*, **181**(12), pp. 2441–2454 (2011).
 60. Karimi, N., Zandieh, M., and Karamooz, H.R. “Bi-objective group scheduling in hybrid flexible flowshop: A multi-phase approach”, *Expert Systems with Applications*, **37**(6), pp. 4024–4032 (2010).
 61. Jolai, F., Asefi, H., Rabiee, M., et al. “Bi-objective simulated annealing approaches for no-wait two-stage flexible flow shop scheduling problem”, *Scientia Iranica*, **20**(3), pp. 861–872 (2013).
 62. Taguchi, G., *Introduction to Quality Engineering: Designing Quality into Products and Processes*, No. 658.562 T3 (1986).
 63. Rahmati, S.H.A., Hajipour, V., and Akhavan Niaki, S.T. “A soft-computing Pareto-based meta-heuristic algorithm for a multi-objective multi-server facility location problem”, *Applied Soft Computing*, **13**(4), pp. 1728–1740 (2013).

Biographies

Naeem Aghajani-Delavar is currently a PhD student at the Department of Industrial Engineering, Islamic Azad University, Qazvin Branch, Iran. He obtained his MSc degree from Islamic Azad University, Qazvin Branch in 2008 and his BSc degree from Iran’s University of Science and Technology in 2006, all in

Industrial Engineering. His research interests are in the areas of operation research such as production planning, production scheduling, fuzzy sets, and meta-heuristic algorithms. He has several papers in journals and conference proceedings.

Esmail Mehdizadeh is currently an Associate Professor at the Department of Industrial engineering, Islamic Azad University, Qazvin Branch, Iran. He received his PhD degree from Islamic Azad University, Science and Research Branch, Tehran in 2009, MSc degree from Islamic Azad University, South Tehran Branch in 1999, and BSc degree from Islamic Azad University, Qazvin Branch in 1996, all in Industrial Engineering. His research interests are in the areas of operation research such as production planning, production scheduling, fuzzy sets, and meta-heuristic algorithms. He has several papers in journals and conference proceedings. Also, he is a Managing Editor of International Journal of Optimization in Industrial Engineering.

Reza Tavakkoli-Moghaddam is a Professor of Industrial Engineering at College of Engineering, University of Tehran in Iran. He obtained his PhD in Industrial Engineering from Swinburne University of Technology in Melbourne (1998), his MSc in Industrial Engineering from the University of Melbourne in Melbourne (1994), and his BSc in Industrial Engineering from the Iran University of Science and Technology in Tehran (1989). He serves as a member in the Editorial Board of five reputable academic journals. He was the recipient of the 2009 and 2011 Distinguished Researcher Awards and the 2010 and 2014 Distinguished Applied Research Award at University of Tehran, Iran. He has been selected as National Iranian Distinguished Researcher for 2008 and 2010. He has obtained the outstanding rank as the top 1% scientist and researcher in the world elite group since 2014. He has published 4 books, 26 book chapters, and more than 1000 journal and conference papers.

Hasan Haleh is currently an Assistant Professor at the Department of Industrial Engineering, Golpayegan University of Engineering, Iran. He received his PhD degree from Gunma University, School of Mechatronics, Japan in 2004, MSc degree from Sharif University of Technology, Tehran in 1994, and BSc degree from Sharif University of Technology, Tehran in 1991, all in Industrial Engineering. His research interests are in automation, multi-criteria decision-making, inventory control, crisis management, change management, and strategic management. He has several papers in journals and conference proceedings.