Soft Computing (2019) 23:9189–9213
https://doi.org/10.1007/s00500-019-03798-7

FOCUS

An efficient hybrid meta-heuristic approach for cell formation problem

Madhu Sudana Rao Nalluri[1] · K. Kannan[2] · Xiao-Zhi Gao[3] · Diptendu Sinha Roy[4]

Published online: 19 February 2019

Abstract

The cellular manufacturing technology, an application of group technology in manufacturing, has been a widely studied combinatorial optimization problem where the entire production system is divided into many cells and part families. In this paper, a novel clonal selection algorithm (CSA) that uses a new affinity function and part assignment heuristic for solving a multi-objective cell formation problem is studied. The proposed CSA has been hybridized with genetic algorithm for generating feasible cell sequences that fulfill both mutual exhaustivity and exclusion properties of machine cells prior to the initial population generation. Additionally, a new part assignment heuristic function that maps parts to machine cells and a novel basic affinity function have been built into the proposed CSA so that it can act as the utility function to solve the multi-objective cell formation problem. This hybrid CSA (HCSA) has been presented and computational results have been obtained for the proposed scheme with a set of 52 benchmark instances collected from literature. The results presented herein demonstrate that overall proposed HCSA is much more promising in comparison with existing approaches available in recent literatures. Extensive statistical and convergence tests have been carried out to ratify the superiority of the proposed HCSA. The improvements can be attributed to the collaborative interactions in the CSA mechanism, the proposed hybridization for initial population generation and so forth.

Keywords Clonal selection algorithm · Cell formation problem · Hyper-mutation · Grouping · Genetic algorithms

1 Introduction

The long-term productivity of a manufacturing unit significantly depends on its underlying design optimization strategy. But one of the fundamental challenges associated with the production environment is the minimization of production cost without compromising product quality. Some popular techniques used to accomplish the aforesaid problem include minimization of flaws in the production process, inefficient use of manpower, transit time, and setup time (Wemmerlöv and Hyer 1987; Wemmerlov and Johnson 1997).

During any manufacturing process, machines are employed to move parts from one place to another; hence, such movements give rise to additional costs. At the same time, frequent variations in production orders can be noticed in small- and medium-scale industries, thus resulting in frequent setup changes and machine movements.

In this respect, group technology (GT) is a method that addresses these problems by introducing a more flexible manufacturing process known as manufacturing cell design (Chan et al. 2006). Group technology (GT) identifies and

Communicated by A. K. Sangaiah, H. Pham, M.-Y. Chen, H. Lu, F. Mercaldo.

✉ Diptendu Sinha Roy
diptendu.sr@nitm.ac.in

Madhu Sudana Rao Nalluri
madhu031083@gmail.com

K. Kannan
kkannan_1960@yahoo.com

Xiao-Zhi Gao
xiao.z.gao@gmail.com

[1] School of Computing, SASTRA Deemed University, Thanjavur, Tamilnadu, India

[2] School of Humanities and Sciences, SASTRA Deemed University, Thanjavur, Tamilnadu, India

[3] School of Computing, University of Eastern Finland, Kuopio, Finland

[4] Department of Computer Science and Engineering, National Institute of Technology Meghalaya, Shillong, Meghalaya, India

creates manufacturing cells in which a group of machines can operate a product family with little or no inter-cell movements. GT was first introduced by Mitranov (1959) and later publicized by Burbidge (1971). The design procedure of a manufacturing cell comprises the following three main phases: (1) grouping of machines as machine cells, also known as the cell formation problem (CFP), (2) arrangement of parts into different part families depending on the production similarities, and (3) mapping of these machine cells and product families (McAuley 1972; King 1980; Ham et al. 1985). According to the literature (Dimopoulos and Zalzala 2000), CFP is one of the proven NP-hard problems.

1.1 Related work

A wide variety of heuristics and meta-heuristics have been applied to solve the CFP in the past two decades. Selim et al. (1998) have done a thorough review on the cell formation problem in group technology, giving a mathematical formulation of the same, and presented the various techniques employed to solve the same till 1995 (Lin et al. 1996). A number of coefficient-based methods to solve the CFP till the last decade have been reviewed in Yin and Yasuda (2005) which also introduces a novel taxonomy of the CFP based on similarity coefficients. A detailed examination of 52 methods for solving the CFP in terms of objective functions, constraints, solution approaches, computational results, and implementation tools has been presented in Papaioannou and Wilson (2010).

The authors recommended the introduction of multi-objective optimization to account for the dynamic nature of the CFP. Most of the engineering problems, including the CFP, require concurrent optimization of several design objectives with conflicting requirements among them and are popularly termed as a multi-objective optimization problem (MOOP) (Nedjah and Mourelle 2015; Lu et al. 2014). A multi-objective integer programming problem has been formulated to solve the CFP using genetic algorithm (GA) in (Solimanpur et al. 2004). A MOOP developed for the CFP by considering other parameters like machine redundancy, production volume, cost of machines, and sequence of operations. This MOOP had been converted into a single objective using fuzzy programming approach and thereafter solved using GA. The obtained results were compared with NSGA-II results (Saeidi et al. 2014).

Another attempt to solve the dynamic multi-objective cell formation problem (MCFP) has been studied in Bajestani et al. (2009), and its results were compared to SPEA and NSGA-II. A new mathematical model proposed for dynamic cell formation by considering inter- and intra-cell movements of all parts. A simulated annealing (SA) technique was embedded in branch and cut to solve the problem, and the obtained results were compared with the outcome of the standard branch and cut (Dalfard 2013).

Albadawi et al. (2005) presented a two-phase mathematical approach for solving the CFP where machine cells were identified by applying factor analysis on the similarity coefficient matrix and after that in phase 2 parts were to identified machine cells using integer programming (Albadawi et al. 2005). A multi-functional mathematical model was also formulated for solving the CFP where the numerical examples were solved by linear interactive and discrete optimizer package (Tsai and Lee 2006).

References (Oliveira et al. 2009; Selim et al. 2003) present various clustering techniques that have been employed to solve the CFP. Similarly, Arkat et al. in 2012 offer a variety of branch and bound algorithms to solve the CFP (Arkat et al. 2012). The CFP under dynamic and uncertain constraints addressed using a fuzzy programming approach. Dynamic constraints exist due to a variable product mix and demand over time. They present a mixed-integer programming model for the dynamic CFP. Since the coefficients of the objective functions are fuzzy numbers, fuzzy programming is used to solve this problem (Safaei et al. 2008).

The primary goal of the model proposed in Arikan and Güngör (2005) includes cell formation and reduction of exceptional elements. This model is capable of expressing vagueness of system parameters and also gives alternative solutions. Reference (Lin et al. 1996) presents a heuristic function based on processing times and demand rates to form the production cells to minimize the cost of inter-cell part movements and intra-cell processing while balancing the workload within each cell. References (Yin and Yasuda 2005; Oliveira et al. 2008) provide reviews on similarity coefficient-based techniques for the CFP.

Several artificial neural network (ANN) techniques have been employed to solve the CFP, and few researchers have used binary incidence matrix as the input parameter for training the ANN. Some authors have taken both binary incidence matrices, and the similarity metric parameter values were derived using similarity coefficients which were subsequently used to train ANN. Variants of ANNs employed in the literature to solve the CFP include back-propagation neural network, Hopfield neural network, adaptive resonance theory (ART), ART1 and so forth (Chu 1997; Chattopadhyay et al. 2012; Chen and Cheng 1995; Guerrero et al. 2002; Pandian and Mahapatra 2009; Yang and Yang 2008).

Several evolutionary techniques inspired by nature have been employed in the literature to solve the single- and multi-objective versions of the CFP, namely genetic algorithm (Onwubolu and Mutingi 2001; Hwang and Sun 1996; Gonçalves and Resende 2004; Mahdavi et al. 2009), hybrid genetic algorithm (James et al. 2007), ant colony optimization (Li et al. 2010), simulated annealing (Pailla et al. 2010) and so on (Wu et al. 2010).

Recent literatures indicate the limitations of previously published works in terms of under-quality solutions, increased hardness with size of the data, generation of infeasible solutions etc.

In Mahmoodian et al. (2017), a new variation of particle swarm optimization is introduced to solve cell formation problem. The swarm intelligence of PSO and artificial intelligence of self-organization map neural networks are combined to solve the cell formation problem. Thirty test instances and a real-time case study are tested using the proposed technique. Noktehdan et al. (2016) have developed a meta-heuristic algorithm to solve a class of grouping problems (bin packing, graph coloring, scheduling, etc.). To test the efficacy of the proposed algorithm, 35 benchmark test instances are tested. The proposed league champion algorithm reached the best solution for 29 of the 35.

The cell formation problem has been solved by combining simulated annealing (SA) and genetic algorithm (GA) in Zeb et al. (2016). The exploration power of GA and intensification power SA have been combined to solve CFP. Boltzmann probability distribution functions and shake-offs are used to overcome from local minimums. Thirty-five benchmark problems are tested to check the efficacy of the proposed method. It has shown improvement in two test instances. In Rajesh et al. (2017), similarities and dissimilarities of a cell are considered to solve cell formation problem. The performance measures used in this work include percentage of exceptional elements, machine utilization, grouping efficacy and cell efficiency. A summary of recent techniques employed to solve the MCFP along with their characteristics, advantages and limitations is summarized and given in Table 1.

In the above all techniques, there is a possibility of generating infeasible solutions probably stuck with local optimum. In evolutionary algorithms research, it has been quite common to introduce hybrid (or) modified evolutionary algorithms to achieve global optimum solutions (Cui et al. 2018; Cai et al. 2016, 2018; Zhang et al. 2018; Cui et al. 2017; Abdel-Basset et al. 2017a, b, 2018a, b; Balakrishnan and Sangaiah 2017a, b; Chiang et al. 2018; Rao et al. 2018; Nalluri and Roy 2017; Srikanth et al. 2018; Verma et al. 2017; Zheng et al. 2018). In this manuscript, we are always aiming to generate feasible solutions with good exploration of the solution space. So to achieve it, the clonal selection algorithm has been hybridized with very simple genetic algorithm and part family assignment heuristic to solve multi-objective cell formation problem (MCFP). The novelties of our proposed approach are threefold: (1) introduction of a new affinity function to meet the needs of optimization under clonal selection, (2) presentation of a new heuristic function to assign the part to a machine cell, and (3) hybridization of GA and CSA to generate a feasible cell sequence that is characterized by mutual exclusivity among the machine cells. Solving a MCFP using any evolutionary algorithm involves the checking of hard constraints every time. But in the proposed CSA-GA hybrid method, feasible cell sequences are generated using a genetic algorithm with less computational effort, and then the machine cells are framed considering these cell sequences. Moreover, swap mutation operation is employed in CSA to produce new solutions; this eliminates the checking of hard constraints related to machine cells.

The remaining paper is organized as follows: Sect. 2 depicts the basic CFP and the related assumptions, and the examples of combinatorial nature of our problem. The objective functions and constraints are also presented in this section. Section 3 introduces CSA with an elaborate discussion on the mapping of MCFP to CSA. The hybridization of CSA with GA and the part assignment heuristic approach have been delineated in Sect. 4. In Sect. 5, the results obtained from the simulations are discussed. Finally, Sect. 6 concludes the study by signalizing the possible future research directions.

2 The multi-objective cell formation problem (MCFP)

The cell formation problem can be represented as a binary incidence matrix in which its rows represent the machine features and columns signify the part dependencies. A solution to the CFP entails the rearrangement of rows and columns of the said matrix to arrive at a block diagonal matrix. Such rearrangements are governed by minimum inter-cell movements and better machine utilization among the cells.

In the said incidence matrix, value one at location (i, j) depicts the involvement of ith machine in the jth part production. Similarly, the value 0 at position (m, n) indicates that mth machine is not involved in the production of mth part. For example, Fig. 1a depicts a CFP with five machines and five parts and their corresponding dependencies are represented by the binary values. After an appropriate rearrangement of the matrix, Fig. 1b was obtained. It is observed from Fig. 1b that no 1's and 0's are present in the non-diagonal and diagonal blocks, respectively; it implies that there are no idle machines and inter-cell movements. Therefore, it can be said that Fig. 1b represents the best CFP solution of Fig. 1a. The rearrangement involved the grouping of machines and parts as follows: cells $\{C1, C2\} = \{\{2, 4\},\{1, 3, 5\}\}$ and parts $\{P1, P2\} = \{\{5, 2, 4\},\{1, 3\}\}$. The mapping between machine

Table 1 Summary of recent techniques used to solve various cell formation problems

Sl. nos.	Technique proposed	Characteristics	Advantages	Limitations
1.	Unsupervised network architecture based on interactive activation and competition (Chu 1997)	A good alternative decision support tool	Can simultaneously form machine cells and part families	–
2.	Self-organizing map(SOM) and visual clustering properties (Chattopadhyay et al. 2012)	Visual clustering approach by unsupervised learning	SOM map size can be decided based on the errors (quantization error, topography error) and average distortion measure	Computational time taken for the large datasets is relatively higher
3.	Self-organizing neural network and linear network flow model (Guerrero et al. 2002)	Unsupervised learning, with computation of weighted similarity coefficients	A systematic two-phase method to create part families and machine cells and inherent parallelization	Parallelization of capability not addressed
4.	Clustering using adaptive resonance theory (ART) and a new performance metric for performance verification (Pandian and Mahapatra 2009)	Clustering methodology by ART	Converted non-binary data into binary data and reduction of computational time	Machine capacity, production volume, layout considerations not been considered
5.	Genetic algorithm(GA) (Onwubolu and Mutingi 2001)	Statistical optimization technique inspired by the nature	Simultaneous grouping of machines and parts and reasonable computational time	Convergence of GA is not proved
6.	Genetic algorithm(GA) (Hwang and Sun 1996)	Two-phase heuristic (GA and greedy heuristic)	A systematic two-phase method to create part families and machine cells and inherent parallelization	No performance and convergence analysis
7.	GA and local search heuristic (Gonçalves and Resende 2004)	Combination of local search heuristic and GA	A random keys alphabet, elitist selection strategy and parameterized uniform crossover	Performance is not validated/convergence is not proved
8.	Genetic algorithm (Mahdavi et al. 2009)	Modified genetic algorithm	Voids and exceptional elements are minimized by finding dedicated manufacturing cells	Performance has not been validated/convergence is not proved
9.	Modified ART1 (Yang and Yang 2008) with relation similarities between parts	Robust and adaptive	The vigilance parameter is estimated based on the data	No performance validation
10.	Multi-objective matrix-based bacteria-foraging optimization algorithm (Nouri 2016)	Non-gradient optimization	Chemotaxis, swarming, reproduction, and elimination dispersal	No performance validation/convergence proof
11.	CPLEX solver (Sakhaii et al. 2016)	A robust optimization approach for solving mixed-integer linear programming	Handled dynamic cellular manufacturing system with unreliable machines	Solutions of linear and nonlinear models are not compared
12.	Parallel simulated annealing (Su and Hsu 1998)	Adaptive, flexible, and efficient	Can be used to solve real multi-part cell formation problem	No discussion of results validation
13.	Genetic algorithm (Sahin and Alpay 2016)	Stochastic optimization technique inspired by the nature	Cubic cell formation problem with human-related issues	Missing of repair functions to convert infeasible solutions into feasible solutions
14.	Multilayer perceptron (MLP), SA, ant colony optimization and branch and bound (Delgoshaei and Gomes 2016)	Supervised learning and combinatorial optimization	Handled unreliable machines and uncertain system cost	Time-consuming process due to back-propagation technique

cells and part families is denoted as $\{C1 \leftarrow P1, C2 \leftarrow P2\}$.

Though the best solution for example 1 was obtained by the proper arrangement of rows and columns, it is not the case for all CFPs. For some CFPs, it is possible that even after rearrangement, 0's might remain in the diagonal blocks and 1's in the non-diagonal blocks. Figure 2a illustrates an example of such CFP problem. Figure 2a describes a CFP problem with five machines and five parts, and their corresponding dependencies are represented by the binary values. After rearranging the matrix, Fig. 2b is obtained. It can be observed from Fig. 2b that four 1's are present in the non-diagonal blocks, where a 0 occupies a diagonal block. This denotes that there is one idle machine

MACHINES	PARTS	1	2	3	4	5
	1	1	0	1	0	0
	2	0	1	0	1	1
	3	1	0	1	0	0
	4	0	1	0	1	1
	5	1	0	1	0	0

(a) A simple binary machine-part incidence matrix

MACHINES	PARTS	5	2	4	3	1
	4	1	1	1	0	0
	2	1	1	1	0	0
	3	0	0	0	1	1
	1	0	0	0	1	1
	5	0	0	0	1	1

(b) Optimal block diagonal Matrix for 1(a)

Fig. 1 **a** A simple binary machine part incidence matrix and **b** optimal block diagonal matrix for **a**

while other machines are working and also four inter-cell movements exist. The rearrangement that involved the grouping of machines and parts is as follows: {*C*1, *C*2} = {{2, 5},{4, 3, 1}} and {*P*1, *P*2} = {{1, 3, 4},{2, 5}}. The mapping between machine cells and part families can be denoted as {C1 ← P1, C2 ← P2}.

These non-diagonal 1's in Fig. 2b are called exceptional elements, and the 0's in the diagonal blocks are called voids. Exceptional elements and voids signify the inter-cell movements of parts and machine idle times during their manufacturing process, respectively. The total number of exceptional elements and that of voids in Fig. 2b are 4 and 1, respectively, so four inter-cell movements exist in the best solution of Fig. 2a. In general, an optimal result of an MCFP is desired to satisfy the following conditions:

1. Minimization of the number of exceptional elements (EE).
2. Minimization of inter-cell movements (ICM).
3. Minimization of the number of Voids (V).

Hence, based on the aforesaid optimal conditions, Fig. 1b is the best solution of Fig. 1a and Fig. 2b is the best option of Fig. 2a.

During solving an MCFP problem, three objectives should be minimized. The first objective is to minimize the number of exceptional elements to reduce the part manufacturing dependency over the cells. It should be understood that a minimum number of exceptional elements do not always signify minimum part movements. The second objective is to minimize the number of inter-cell movements of a part to reduce the transportation time and production cost. The minimization of these two objectives leads to the minimization of manufacturing time of a part, but the minimization of the first objective does not always minimize the second due to the dependencies of part manufacturing and dissimilarity among the parts and machine cells. Therefore, these two objectives are not always proportional. The third objective is to minimize the number of voids to reduce machine idle time for better machine utilization. The following index sets, parameters, and variables are used in the model formulation:

Index sets	Parameters	Variables
i: Used for parts	*m*: Total number of machines.	*C*: Set of cells.
	n: Total number of parts	|*C*|: Total number of cells
j: Used for machines	*M*: Set of machines	C_k: Represents *k*th cell. $\|C_k\|$: Total number of machines in cell number *k*
	P: Set of parts	
	$M_{\min:}$ Minimum number of machines in each cell	P_k: Represents *k*th part family
k: Used for cells	$M_{\max}$: Maximum number of machines in each cell	$\|P_k\|$: Total number of parts in the part family *k*
	$x_{ij} = \begin{cases} 1 & i\text{th part production depends on } j\text{th machine} \\ 0 & \text{otherwise} \end{cases}$	$y_{jk} = \begin{cases} 1 & j\text{th machine belongs to } k\text{th cell} \\ 0 & \text{otherwise} \end{cases}$
		$p_{jk} = \begin{cases} 1 & i\text{th part belongs to } k\text{th part family} \\ 0 & \text{otherwise} \end{cases}$
		$C_k(i,j) = \begin{cases} 1 & i\text{th part production depends on } j\text{th machine and also } (i,j) \text{ position falls in } k\text{th cell} \\ 0 & \text{otherwise} \end{cases}$

	PARTS					
MACHINES		1	2	3	4	5
	1	1	1	0	0	1
	2	1	0	1	1	1
	3	0	1	0	1	1
	4	1	0	0	0	1
	5	1	0	1	1	0

(a) A simple binary machine-part incidence matrix

	PARTS					
MACHINES		**1**	**3**	**4**	**2**	**5**
	5	1	1	1	0	0
	2	1	1	1	0	1
	4	1	0	0	0	1
	3	0	0	1	1	1
	1	1	0	0	1	1

(b) Optimal block diagonal matrix for 2(a)

Fig. 2 **a** A simple binary machine part incidence matrix and **b** optimal block diagonal matrix for **a**

$$\text{Block diagonal sum can be expressed as : } \sum_{K=1}^{|C|} \sum_{j \in C_k} \sum_{i \in P_k} x_{ij} \tag{1}$$

The first objective can be expressed by (2):

$$\text{Objective 1(EE)} = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} x_{ij} \right) - \left(\sum_{K=1}^{|C|} \sum_{j \in C_k} \sum_{i \in P_k} x_{ij} \right) \tag{2}$$

Similarly, the second and third objectives can be expressed by (3) and (4), respectively:

$$\text{Objective 2 (ICM)} = \sum_{i=1}^{n} \sum_{k=1}^{|C|} \bigvee_{j=1}^{m} \left(x_{ij} \wedge y_{jk} \right) \tag{3}$$

$$\text{Objective 3}\,(V) = \left(\sum_{k=1}^{|C|} |C_k| * |P_k| \right) - \left(\sum_{K=1}^{|C|} \sum_{j \in C_k} \sum_{i \in P_k} x_{ij} \right) \tag{4}$$

The constraints related to the three objectives can be expressed as follows:

$$C_i \cap C_j = \emptyset \quad \forall i \neq j \tag{5}$$

$$P_i \cap Pj = \emptyset \quad \forall i \neq j \tag{6}$$

$$C = C_1 \cup C_2 \cup \cdots \cup C_k; \tag{7}$$

$$P = P_1 \cup P_2 \cup \cdots \cup P_k; \tag{8}$$

$$M_{\min} \leq |C_k| \leq M_{\max} \tag{9}$$

$$C_k \subseteq \{1, 2, 3, \ldots, m\}; \quad 1 \leq k \leq |C|$$
$$P_k \subseteq \{1, 2, 3, \ldots, n\}; \quad 1 \leq k \leq |C|$$

Constraint (5) ensures that each machine belongs to exactly one cell, constraint (6) verifies that each part belongs to exactly one part family, and constraint (9) furnishes the lower and upper bounds of the capacity of every cell. Constraints (5) and (7) together represent that all cells are mutually exclusive and exhaustive subsets of C. Constraints (6) and (8) together represent that all part families are mutually exclusive and exhaustive subsets of *P*. The aforesaid constraints are rewritten in terms of machine cell and part cell binary incidence matrices and can be presented by the following equations:

$$\sum_{k=1}^{|C|} p_{ik} = 1 \quad \forall i \tag{10}$$

$$\sum_{k=1}^{|C|} y_{jk} = 1 \quad \forall j \tag{11}$$

$$\sum_{i=1}^{n} \sum_{k=1}^{|C|} p_{ik} = n \tag{12}$$

$$\sum_{j=1}^{m} \sum_{k=1}^{|C|} y_{jk} = m \tag{13}$$

$$\sum_{j} C_k(i,j) \geq 1 \quad \forall k \quad \text{and} \quad i \tag{14}$$

$$\sum_{i} C_k(i,j) \geq 1 \quad \forall k \quad \text{and} \quad j \tag{15}$$

It is important to notice that there should be at least one dependent machine and one dependent part in the *k*th machine cell, so for each row and column of an MCPF problem, at least one 1 should be present in the rows and columns of binary sub matrices. The constraints (14) and (15) guarantee it. The 0 rows and columns of MCPF signify that there is no work for the corresponding part and machine in that machine cell and part family; therefore, the corresponding machine and part must be removed from the corresponding machine cell and part family.

In order to make assessments on the quality rearrangements of the machine parts incidence matrix of a CFP, a number of measures have been proposed in the literature. One of them is group efficiency (Chandrasekharan and Rajagopalan 1986b), η, which can be expressed as follows:

$$\eta = q\,\eta_1 + (1 - q)\,\eta_2 \tag{16}$$

where η_1 is the ratio between the number of 1's in the diagonal blocks of the solution matrix and the total number of elements in the diagonal blocks and η_2 is the ratio between 0's in the off-diagonal blocks of the solution matrix and total number of elements in the off-diagonal blocks.

Despite its extensive use, it is found that η loses its discrimination for certain matrix sizes. So an alternative measure known as group efficacy (Eq. 17) has been proposed later (Suresh Kumar and Chandrasekharan 1990).

$$\text{Group Efficacy, GE} = \frac{e - e_0}{e + e_v} \tag{17}$$

where e represents the total number of 1's in the machine parts incidence matrix, e_0 is the total number of 1's in the off-diagonal blocks of block diagonal matrix, and e_v is the total number of voids in the block diagonal matrix.

3 Clonal selection algorithm for the MCFP

In this paper, MCFP problem was solved using clonal selection algorithm (CSA) and meta-heuristic algorithm. Section 3.1 introduces the CSA, and Sect. 3.2 describes the use of CSA in the MCFP using a novel affinity function to rank candidate solutions. It also presents a novel heuristic function for finding out the part families. In Sect. 4, CSA is hybridized using GA to generate a feasible cell sequence as initial population.

3.1 Essential steps of clonal selection algorithm

CSA is one of the biologically inspired techniques introduced by (De Castro and Von Zuben (2002). CSAs can be used for pattern recognition and function optimization problems (White and Garrett 2003). Several variations of the CSA have been introduced to solve engineering optimization problems, such as economic load dispatch problem (Panigrahi et al. 2007), job shop scheduling problem (Yang et al. 2008), and network intrusion detection (Kim and Bentley 2001).

Clonal selection algorithm replicates the immune systems of certain vertebrates, while some detectors (for detecting new pathogens like viruses) identify new intruders in the body. Detectors are characterized by the mechanism to improve recognition performance using an iterative process known as affinity maturation (Floreano and Mattiussi 2008). It is assumed that the system has a number of receptors and each of them is basically a candidate solution.

The first fundamental step in CSA is to define the structure of receptors to represent the problem solutions. Based on the structures of receptors, a set of initial populations (R) has to be generated either randomly by following a particular distribution or by employing some heuristics. An affinity function (AF) has to be developed to determine the strength of the solution. Based on the AF values, some receptors with high-affinity values are selected from R and denoted as R_H. The set of R_H is expanded by applying the cloning method, and the number of clones is determined by the affinity values of the receptors. A collection of receptors, R_M, are generated by applying the appropriate mutation operators over R_H. Finally, a set of best solutions are selected from the sets R, R_H, R_R, and R_M based on their objective function values. The above process is repeated iteratively several times until a pre-defined termination condition is reached. The termination conditions may either be a pre-defined number of iterations or some pre-defined solution accuracy.

3.2 Mapping of MCFP as a CSA

3.2.1 Initialization of parameters

In this step, the termination criterion and also various parameters, such as the sizes of R, R_H, R_R, and R_M, are initialized. In this paper, a number of iterations (50) were used as the termination criterion. Also, the choice of parameters, such as mutation probability, the expected distribution of random numbers, etc., are initialized in this step. The relation between the affinity value of a receptor and the number of clones applied on that receptor is also defined in this step.

3.2.2 Machine cell and part family encodings

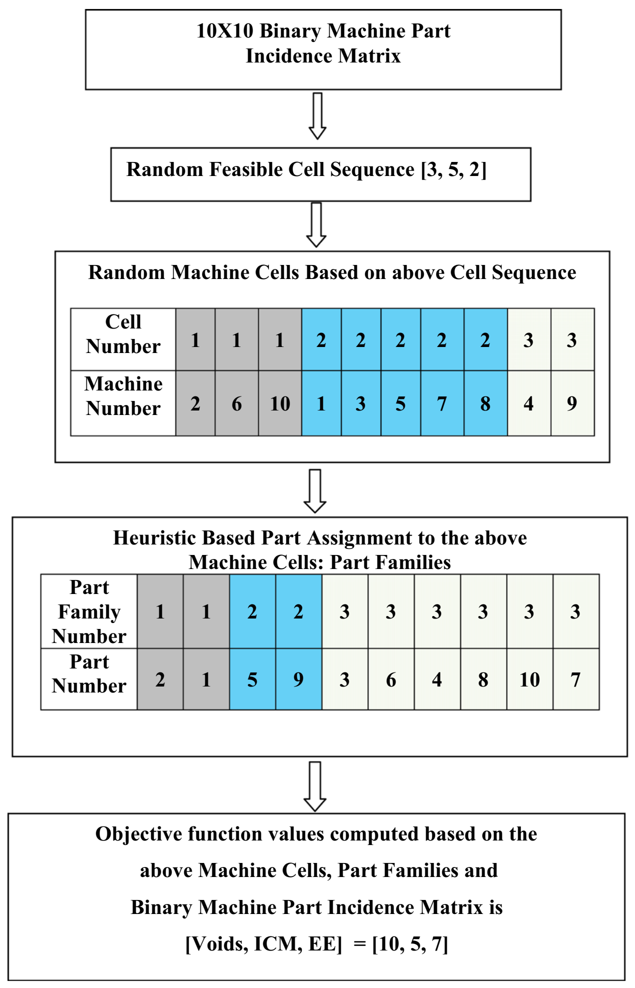
Each machine belongs to exactly one cell, and each part belongs to exactly one part family. Machine cells are encoded as a vector, namely Machine_Vector (1, j), and it signifies that the jth machine belongs to the kth cell. This vector is also used to prepare the machine cell incidence matrix. For instance, let us consider the following machine cell vector of three machine cells:

2	1	2	3	2	1	2	2	3	1

Cell-1 = {2, 6, 10}, Cell-2 = {1, 3, 5, 7, 8}, Cell-3 = {4, 9}. The corresponding machine cell incidence matrix can be formulated as follows: The first entry of the vector is 2; therefore, in the machine cell incidence matrix, the value of first row and second column is 1 and the rest of the values in this column are 0's.

The part family is also encoded as a vector of cell numbers, namely Part_Vector (1, t) = p, and it means that a part belongs to the pth part family. For example, let us consider the following machine cell vector of three part families:

1	2	3	2	2	1	2	3	2	1


10X10 Binary Machine Part Incidence Matrix

Random Feasible Cell Sequence [3, 5, 2]

Random Machine Cells Based on above Cell Sequence

Cell Number	1	1	1	2	2	2	2	2	3	3
Machine Number	2	6	10	1	3	5	7	8	4	9

Heuristic Based Part Assignment to the above Machine Cells: Part Families

Part Family Number	1	1	2	2	3	3	3	3	3	3
Part Number	2	1	5	9	3	6	4	8	10	7

Objective function values computed based on the above Machine Cells, Part Families and Binary Machine Part Incidence Matrix is [Voids, ICM, EE] = [10, 5, 7]

Fig. 3 Machine cell and part family encodings of a 10 × 10 machine part binary incidence matrix example

PF-1 = {1, 6, 10}, PF-2 = {2, 4, 5, 7, 9}, PF-3 = {3, 8}. The corresponding part family incidence matrix is formulated as follows: The first entry in the vector is 1; therefore, in the part family incidence matrix, first row and first column value is '1' and the rest of the values are 0's in that column. The above process is delineated in Fig. 3.

3.2.3 Affinity function

A CSA requires a function that converts the value of optimization function to any affinity value. The affinity of receptors decides the mutation rate, the number of clones, etc.; thus, it is a primary parameter of CSA. In this paper, an affinity function (AF) comprised of the utility function as denoted in (18) was defined:

$$\mathrm{AF} = (1 + \text{Total Intercell moments})^{-(\text{Voids}+\text{TotalnumberofExceptionalElements})} \tag{18}$$

The maximum value of this affinity function reaches 1 when the total number of exceptional elements is zero. The minimum value of this affinity function reaches close to 0 when the values for voids, exceptional elements, and inter-cell movements are large. When two solutions have the same number of inter-cell movements and different numbers of voids, the solution with a lesser number of voids has better AF than the counterpart solutions with more number of voids. It is a nonlinear and monotonically decreasing function of the three objectives, and therefore, it works both as a utility function for solving the MCFP and fuzzy membership function of three objectives. The affinity values of the receptors can be used to rank the receptors, and corresponding Pareto-optimal solutions can be obtained (Deb 2001). This AF decides the number of clones to be applied to a receptor and also ranks the receptors.

3.2.4 Generating the initial population

The initial population of receptors in CSA can be made either randomly by a probability distribution or by developing a heuristic function based on the problem-specific information. Before forming the machine cells, a cell sequence should be found out to specify the number of machines in each cell. In an MCFP with C number of cells, the minimum and maximum constraints are represented as $M_{\min}$ and $M_{\max}$, and the size of the cell sequence space is denoted as $(M_{\min} - M_{\max})^C$. For instance, if $C = 5$, $M_{\min} = 5$, and $M_{\max} = 11$, then the size of the cell sequence space is 6^5. This space can be explored for an optimal solution through an enumeration process but that will force the system to check the number of infeasible solutions within the space because of the mutually non-exhaustive mapping of cells. Each number in the range of $[M_{\min}, M_{\max}]$ can appear at most C times in the cell sequence. Because of this uniformity, several cell sequences lead to infeasibility. The size of the cell sequence space can be reduced drastically by increasing the appearance limit of a particular number in the range $[M_{\min}, M_{\max}]$. This is enumerated by employing Algorithm 1 as presented hereafter.

```
==================================================================
Algorithm-1: To find the feasible occurrences of a number in the given cell sequence
==================================================================
Cell-Machine-Pattern ($M_{min}$, $M_{max}$, CELL, Machines)
Input Parameters      :$M_{min}$, $M_{max}$, CELL
Output                :N_TS[ ]

// This function returns upper bound on the occurrence of a number $i \in [M_{min}, M_{max}]$ in cell sequence
 1. for i=$M_{min}$ to $M_{max}$
 2.     for k=1 to CELL
 3.         Con1=$M_{min}$*(CELL-i);
 4.         Con2= $M_{max}$*(CELL-i);
 5.         if ( Con1 $\le$ (Machines – (k*i)) $\le$ Con2 ) then
 6.               continue;
 7.         else
 8.               break;
 9.         endif;
10.     endfor;
11.     if (k<CELL) then
12.           P=k-1;
13.     else
14.           P=K;
15.     endif;
16.     N_TS[i]=P; // number i, can occur P times in the cell sequence.
17.  endfor;
18. return N_TS;
19. end
==================================================================
```

3.2.5 Mutation operation

New receptors are evolved by the application of various mutation operations on existing receptors. The different types of mutation operations in biology include complementation, inversion, substitution, insertion, deletion, reciprocal recombination, and nonreciprocal recombination (Floreano and Mattiussi 2008). All these mutation operations are not possible to apply to construct R_M from R_H due to the possibility of violation of the hard constraints, namely (5), (6), (7), and (8). Moreover, the total number of cells increases due to the insertion operations; therefore, the only applicable mutation operation in CSA is the swap mutation operation. During the execution of swap mutation operation, values are exchanged between different locations, so the machine membership of a cell only changes without violating the aforesaid hard constraints. For example, if we consider a receptor with ten positions and apply swap mutation between positions 1 and 4, then the positions of the original receptor and mutated receptor will be as Fig. 4a, b, respectively.

4 The hybrid clonal selection algorithm

In the basic CSA, random generation of receptors through an enumeration process yields several infeasible solutions; therefore, a lot of computations go in vain or additional computations have to be spent to convert those infeasible solutions into feasible one. So the performance of CSA can be improved considerably using some appropriate heuristics or meta-heuristic approaches. To accomplish this task, a collection of feasible cell sequences which satisfy Eqs. (5), (7), and (9) have to be formulated before generating the machine cells. In this regard, this paper proposes a hybrid clonal selection algorithm (HCSA) by combining genetic algorithm with the basic CSA, and the corresponding system diagram is given in Fig. 5. The role of genetic algorithm in HCSA is to generate feasible cell sequences as exhibited in Fig. 6. In the proposed HCSA presented in Algorithm 2 (Fig. 7), there were five phases with a dedicated functionality of each phase. In phase 1, all problem-specific parameters and algorithmic parameters were initialized. In phase 2, a GA was executed to generate the feasible cell sequences. In later phases, these sequences were used to generate random feasible machine cells, and

2	1	2	3	2	1	2	2	3	1

(a) Before Swap mutation

3	1	2	2	2	1	2	2	3	1

(b) After Swap Mutation

Fig. 4 **a** Before swap mutation and **b** after swap mutation

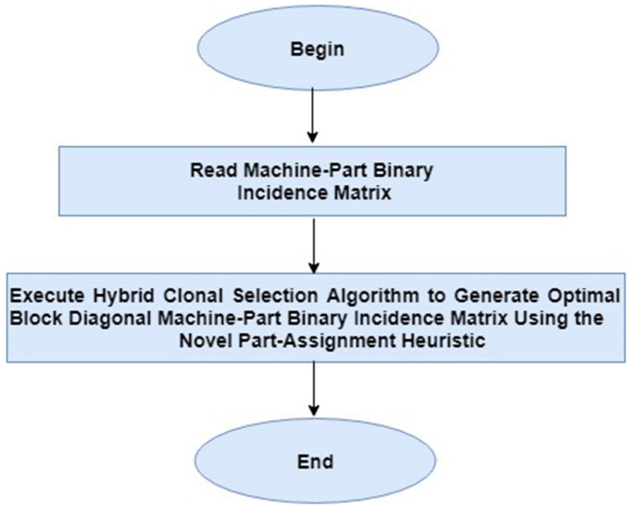

Begin
Read Machine-Part Binary Incidence Matrix
Execute Hybrid Clonal Selection Algorithm to Generate Optimal Block Diagonal Machine-Part Binary Incidence Matrix Using the Novel Part-Assignment Heuristic
End

Fig. 5 System diagram of HCSA

the corresponding steps are mentioned in Algorithm 3. Algorithm 3 was called from step 3 of Algorithm 2.

In the next phase, machine cells (R) were constructed randomly using the cell sequences, and the corresponding steps are mentioned in Algorithm 4. Algorithm 4 was called from step 6 of Algorithm 2. In phase 4, machine cells and part family binary incidence matrices were generated for the receptors in R. Machine cell binary incidence matrix was generated using receptors in R (Algorithm 6), and their corresponding part families were formulated using the novel heuristic function. After that, the three objective functions were computed for the receptors in R, and their affinity value was computed. Phase 5 has a finite loop of statements, and in each iteration of this loop, a collection of R_R, R_M, and R_H was constructed. To formulate R_R, phase 3 was utilized again in phase 5. The construction of R_H from R involved the finite cloning operation and the use of affinity function. Similarly, R_M was constructed from the collection of receptors R_H by applying the swap mutation operations for a limited number of times. At the end of each iteration, based on their affinity values, few receptors were selected from the collection R_R, R_H, and R_M and passed to the next iteration. The affinity function acted as a utility function for the proposed MCFP. The termination condition of the proposed HCSA depended on the number of generations evolved.

The performance of the proposed HCSA is very promising because it avoids the checking of violation of hard constraints for every receptor. It always generates feasible solutions using GA and swap mutation operations. In the proposed HCSA, machine cells were generated randomly through receptors; however, these receptors did not directly affect the part family generations.

Therefore, there should be a mechanism that can efficiently assign a machine cell for each part. In this research, a heuristic function known as part assignment heuristic was developed to address this issue.

```
========================================================================
Algorithm-2: Hybrid_Clonal_Selection_Algorithm
========================================================================
HCSA()
1.   Initialize the parameters: machines, parts, $M_{min}$, $M_{max}$, NN,Cells.
2.   Read MP matrix.  // MP=Machine –Part Binary incidence matrix.
3.   Cell_Pattern =Genetic_Algorithm($M_{max}$,$M_{min}$,machines,Cells,NN)
4.   [r1,c1]=size(Cell_Pattern);
5.   mm=k;  N=$r_1$*mm;
6.   X= Random_Machine_Cell (mm, Machines, Cells)
7.   [Affinity,Obj] =Mc_Pf_affinity(MP,X)
8.   rank =Rank_affinity(affinity)
9.   for i=1 to k
10.          R   = Best_solution(X, rank, t)  // select the t-number of best solutions based on Ranking.
11.          $R_H$ = Best_Affinity_Selection(R,p )
12.          $R_M$ = Swap_Mutation ($R_H$)
13.          $R_R$=  Random_Machine_Cell (p2, Cells, Machines)
14.          X= R ∪ $R_H$ ∪ $R_M$ ∪ $R_R$
         //  preparing Machine-Cell & Part-Family incidence matrices.
         // and computing objective functions and affinity of it.
15.          [Affinity,Obj] =Mc_Pf_affinity (MP,X)
16.          Write [X, Affinity and Objective_values] in to data base
17.          rank = Rank_affinity(Affinity)
18.  endfor i
19.  Print Affinity(rank(1))
20.  end
========================================================================
==========================================================
Algorithm-3: Genetic_Algorithm for generating feasible cell sequences
==========================================================
  Genetic_Algorithm($M_{max}$,$M_{min}$,,machines,Cells,NN)
  Input Parameters  : $M_{max}$, Mmin, Cells, machines,NN
  Output            : machine_cell_pattern.
  // this function generates cell sequence using Genetic Algorithm.
  // only Swap mutation operation is used; Crossover operation is not used.
  //It returns at most NN feasible cell  Sequences
  1.  set pp= $M_{max}$*Cells
  2.  seq=zeros(1,pp);
  3.  seq(1,1:machines)=1
  4.  for i=1 to Max_Iterations
  5.          for l=1 to NN
  6.                  for p=1 to Mutation_ size    // SWAP mutation…
  7.                          pos=rand([1,pp],1,2)   // random number generator..
  8.                          swap= seq(1,pos(1))
  9.                          seq(1,pos(1))= seq(1,pos(2))
  10.                         seq(1,pos(2)) =swap
  11.                 Endfor
  12.         Endfor
  13.         k=0;
  14.         for l=1 to NN
  15.                 status=1;
  16.                 for i=1 to Cells
  17.                         sum1[i]=0;
  18.                         for j=1  to  $M_{max}$
  19.                                 Sum1[i]=Sum1[i]+seq[l,(i-1)*$M_{max}$ + j];
  20.                         endfor
  21.                         if (Mmin $\leqq$ sum1(i) $\leqq$ $M_{max}$ ) // to check the Min and Max constraints
                                    continue ;
  22.                         else
  23.                                 status=0;
  24.                                 break;
  25.                         endif;
  26.                 endfor
  27.                 if ( Status ==1)
  28.                         k=k+1;
  29.                         Machine_cell_pattern (k,:)= sum1;
  30.                 endif
  31.         endfor
  32. return   Machine_cell_pattern.
  33. End
==========================================================
==================================================================
Algorithm-4: Random Generation of Machine Groups
==================================================================
   Random_Machine_Cell (mm , Number_machines, cells)
     1.   XX=Get_cell_sequence ();
     2.   [r_1,c_1]=size(XX) ;
     3.   for l=1 to r_1
     4.     for pp=1 to mm      // mm number of solutions follows each cell sequence
     5.          Lt =[1: Number_Machines];
     6.          for i=1 to cell
     7.               for  z=1  to   XX (l, i)
     8.                     [r,c]   =  size(Lt) ;
     9.                     TU      = randi([1,c],1,1);
    10.                     X((l-1)*mm+ pp, Lt(TU(z))=i ;
    11.                     B =Lt(TU) ;
    12.                     Lt=Set_Difference(Lt,B);
    13.               endfor
    14.          endfor
    15.          Clear Lt ;
    16.     endfor
    17.   endfor
    18.  return   X
   19. end
==================================================================
```

```
==================================================================
Algorithm-5: Preparing Machine-Cell and Part-Family incidence matrices
==================================================================
   Mc_Pf_affinity (MP, X)
   Input Parameters  : MP,X.
   Output            : affinity

   // This function Computes objective function values for the current X
   // Affinity value of solution X is computed using the 3 objective values of X
    1.  for i=1 to N
    2.        bx=[0]_{cells x machines}
    3.        for j=1 to machines
    4.              bx(x(i,j),j)=1;
    5.        endfor j
    6.        Ck= MP * bx'
    7.        P_f  =  Heuristic(Ck,MP,X(i))
    8.        Obj  = Compute_Obj(bx)
    9.        Affinity[i]= $(1 + Obj[2])^{-(Obj[1]+Obj[3])}$
   10.  endfor i
   11.  return [Affinity, Obj]
   12.  end
==================================================================
===========================================================
   Algorithm-6:  Rank Based on Affinity Values
===========================================================
    Rank_affinity(affinity)
    Input Parameter : affinity;
    Output          : rank;
         // this function returns rank of all solutions based on its affinity value.

    1.   for i=1 to N
    2.        [val, pos] =MAX(affinity)
    3.        rank[pos]=i
    4.        affinity[pos]= affinity[pos] - 1
    5.   endfor i
    6.   return rank
    7.   end
===========================================================
```

4.1 Part assignment heuristic

Since the goal of the MCFP is to minimize the number of voids, exceptional elements, and inter-cell movements, the heuristic for part assignment should directly be a function of these three objectives. A part family is a set of parts that are dedicated to a cell in the block diagonal matrix of part-machine binary incidence matrix. The parameters of this heuristic function were derived from the product of part-machine and machine cell binary incidence matrices. The heuristic value of ith part and jth cell was computed using (19):

$$\text{Heuristic } (i,j) = \frac{k_i}{m} + \frac{V_{i,j} + EE_i}{l_{i,j}} \tag{19}$$

where m is the total number of cells, k_i is the number of cell movements made by part i, $V_{i,j}$ is the number of voids caused by ith part in jth cell, EE_i is the number of exceptional elements generated by part i, and $l_{i,j}$ is the number of 1's in the ith column of jth block of the block diagonal matrix. The part family index of ith part was computed using (20):

$$\text{Part\_Family}(i) = \min_j \text{Heuristic}(i,j). \tag{20}$$

4.2 Handling of machine and parts in the machine cells with full of zeros

In the part family choosing problem as mentioned in (20), very rarely it happens for certain machine cells that they are not mapped to any part. In such cases, the machines within these cells have to be merged with other cells. The decision of how to allocate these machines to other cells is again another optimization problem. Such merges might introduce new voids, exceptional elements, and inter-cell movements. One possible approach for solving this sub-problem is to merge each machine with all feasible machine cells and assess the status regarding the increase in the number of voids and the decrease in the number of exceptional elements. Merging of machines to new cells is selected based on the difference between the number of voids newly created and the number of exceptional elements decreased. Moreover, in some cases, few machines and parts violate constraints (14) and (15). In such cases, those machines and parts are allotted to other machine cells and part families based on (19) and (20). Finally, the part assignment heuristic mentioned in Eqs. (19) and (20) can be used to relocate a machine to a machine cell by passing the machine number and other cell numbers as the inputs.

4.3 Time complexity analysis of HCSA

To generate the feasible cell sequences, GA has been executed in line 3 of Algorithm 2 with a clear description given in Algorithm 3. In this algorithm, only mutation operation alone is used and crossover is not used due to the feasibility of generating infeasible cell sequences. There are for loops in lines 4, 5, and 14 of Algorithm 3. Thus, the worst-case time complexity analysis of Algorithm 3 is given below:

$$\begin{aligned}
&(\text{Max\_Iterations} * (NN * \text{Mutation\_size}) + (NN * \text{Cells}))\\
&\quad \le (\text{Max\_Iterations} * (NN * \text{Cells}) + (NN * \text{Cells}))\\
&\qquad (\text{Since Mutation\_size} < \text{ Cells})\\
&\quad = (\text{Max\_Iterations} * 2 * (NN * \text{Cells}))\\
&\quad \cong O(\text{Max\_Iterations} * (NN))\\
&\quad (\text{Since the number of cells is very}\\
&\qquad \text{small compared with NN and Max\_Iterations})
\end{aligned}$$

If the value of 'Max_Iterations' is less than NN, then the time complexity of Algorithm 3 is O(NN^2). Otherwise, it is $O(\text{Max\_Iterations}^2)$. Therefore, the worst-case time

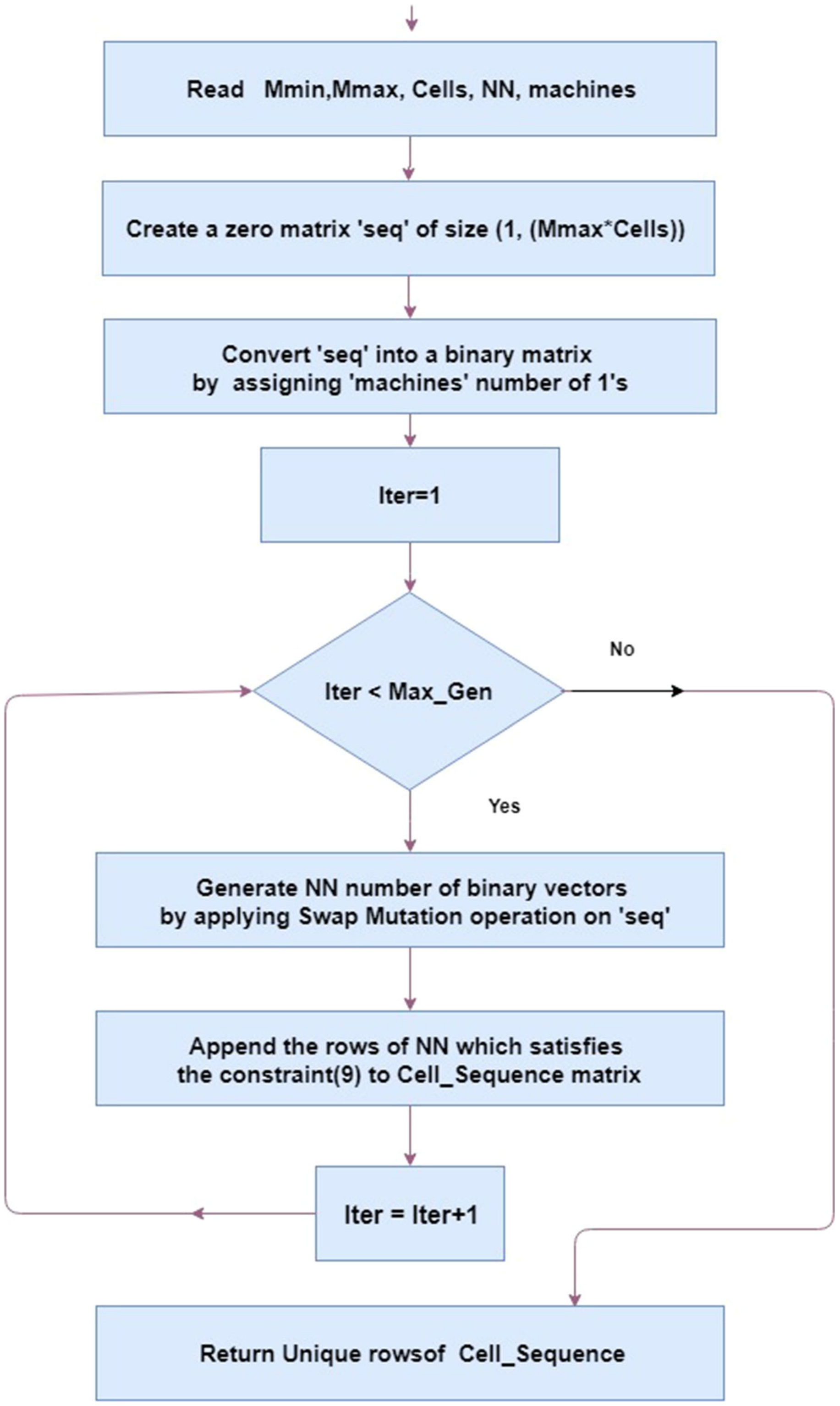

Read Mmin,Mmax, Cells, NN, machines
Create a zero matrix 'seq' of size (1, (Mmax*Cells))
Convert 'seq' into a binary matrix by assigning 'machines' number of 1's
Iter=1
Iter < Max_Gen
No
Yes
Generate NN number of binary vectors by applying Swap Mutation operation on 'seq'
Append the rows of NN which satisfies the constraint(9) to Cell_Sequence matrix
Iter = Iter+1
Return Unique rowsof Cell_Sequence

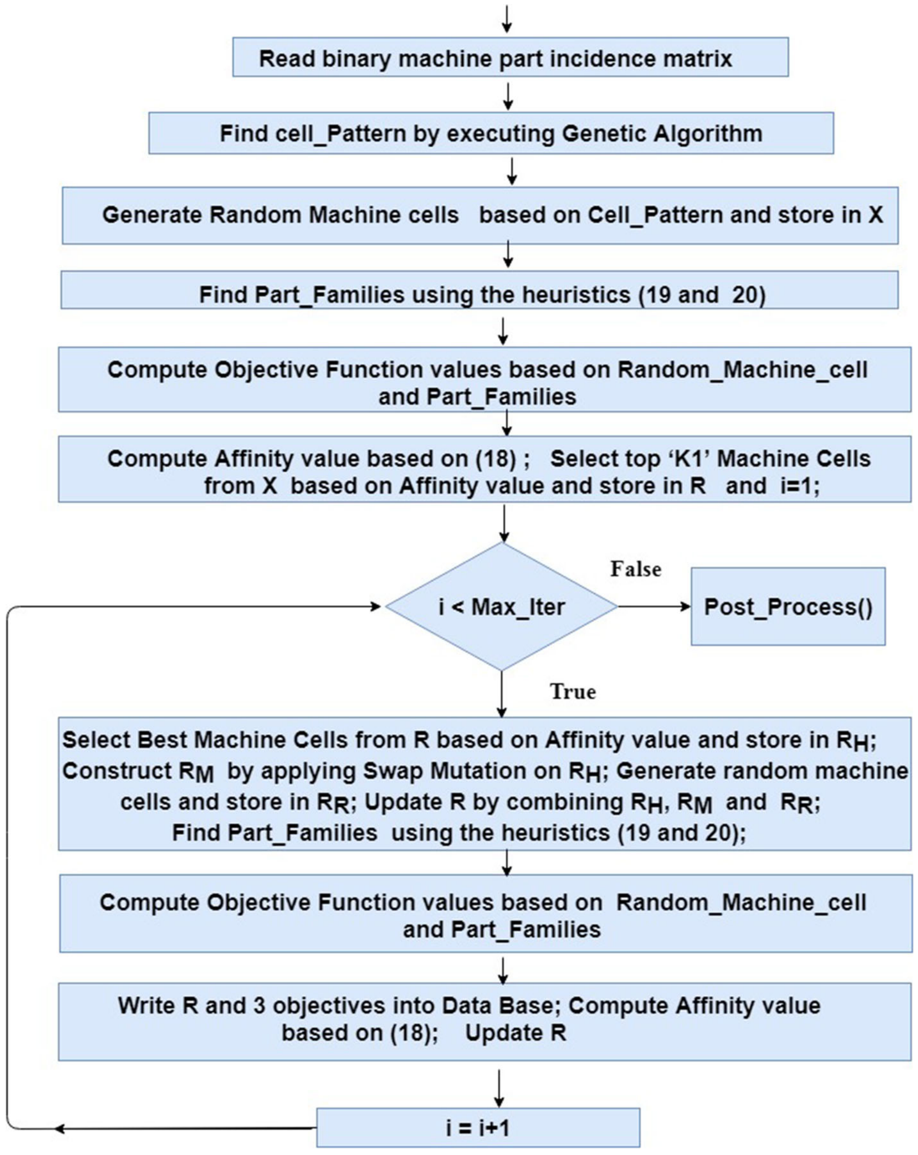
Fig. 6 GA flowchart for generating feasible cell sequences

complexity of Algorithm 3 is a quadratic polynomial in terms of 'Max_Iterations' or '*NN*.'

In line 7 of Algorithm 2, affinity value is computed, which necessitates processing the machine and part binary incidence matrix. The matrix MP is processed for $|X|$ times; therefore, it requires $|X|$*$|MP|$ operations, where $|MP|$ represents the size of the matrix.

In line 8 of Algorithm 2, sorting algorithm is executed on $|X|$ affinity values, so it requires $O(|X^* \lg(X)|)$ number of operations. In line 9 of Algorithm 2, a for loop is executed

Fig. 7 Overall HCSA flowchart


Read binary machine part incidence matrix
Find cell_Pattern by executing Genetic Algorithm
Generate Random Machine cells based on Cell_Pattern and store in X
Find Part_Families using the heuristics (19 and 20)
Compute Objective Function values based on Random_Machine_cell and Part_Families
Compute Affinity value based on (18) ; Select top 'K1' Machine Cells from X based on Affinity value and store in R and i=1;
i < Max_Iter
False
Post_Process()
True
Select Best Machine Cells from R based on Affinity value and store in RH; Construct RM by applying Swap Mutation on RH; Generate random machine cells and store in RR; Update R by combining RH, RM and RR; Find Part_Families using the heuristics (19 and 20);
Compute Objective Function values based on Random_Machine_cell and Part_Families
Write R and 3 objectives into Data Base; Compute Affinity value based on (18); Update R
i = i+1

'k' times. Therefore, in the worst case, it requires (k*(|X|*lg(|X| + |X|*|MP| + |X|)) number of operations. Therefore, finally Algorithm 2 requires:

O(Max_Iterations*(NN) + |X|*|MP| + (k*(|X|*lg(|X| + |X|*|MP| + |X|)) number of operations. Hence, the worst-case time complexity of Algorithm 2 is

$$O((\text{Max\_Iterations} * (NN)) + (|X| * |\text{MP}|) + (k * (|X| * \lg(|X| + |X| * |\text{MP}| + |X|)) \cong O(\text{Max\_Iterations} * (NN) + |X| * |\text{MP}| * (k + 2)) \quad (\text{Since on average } (|X|) < |MP|)$$

By removing all the lower order terms, the worst-case time complexity of Algorithm 2 is:

O(|X|*|MP|*(k)). This value is at most O(|MP|3)or O(|X|3) or O(k^3). Finally, the worst-case time complexity of Algorithm 2 is a cubic polynomial in terms of either number of iterations, or the number of distinct cell sequences, or the size of the machine part binary incidence matrix, whichever is larger.

5 Performance evaluation of the proposed HCSA

Section 5.1 provides the details of the computing environment and other related information about the simulations. Section 5.2 presents the results obtained by the novel affinity function. Sections 5.3 and 5.4 display the statistical test data and performance metrics of HCSA, respectively. Section 5.5 discusses the scalability issues of HCSA.

Finally, Sect. 5.6 discusses the critical observations of the simulation.

5.1 Simulation environment

The execution of the proposed HCSA was done on a desktop personal computer loaded with Intel (R) core$^{(\mathrm{TM})}$ i5-2450M CPU @ 2.50 GHz processor, internal memory of 4 GB (DDR-2), Windows 7 (32-bit), and MATLAB 2013a. Group efficacy was utilized to assess the simulation performance.

5.2 Test datasets and results obtained

The details of the 52 test instances are given in Tables 2, 3, and 4. The results of tests (14, 16, 20, 21, 29, 33, and 34) mentioned in Batsyn et al. (2013) did not satisfy constraints (14) and (15). In the block diagonal matrices of these tests, some machines were grouped with machine cells even though there was no work in that cell, so all entries in that cell were 0's. Also, some parts were transferred to machine cells without any presence of a dependent machine in that cell, so that cell manifested zero column. Hence, these zero rows and zero columns of the block diagonals caused production delay. So to eliminate the presence of zero rows and zero columns in the block diagonals (Batsyn et al. 2013; Bychkov and Batsyn 2018), GE values were modified. The block diagonal matrices were not present in the earlier works (Zeb et al. 2016; Karoum et al. 2018; Mahmoodian et al. 2017; Noktehdan et al. 2016; Pinheiro et al. 2017); therefore, their GE values were not taken into consideration for comparison. In our work, the following criteria were considered to rearrange the zero rows and zero columns: (1) identification of block diagonals that had at least one 1 for previously identified machines and parts, (2) allocation of a machine (part) to one of the machine cells (part family) to condense number of voids and exceptional elements. Finally, the block diagonals of the test instances (14, 16, 20, 21, 29, 33, and 34) did not have zero rows and columns in the block diagonals. Hence, the GE values of these test instances were recomputed, and the updated GE values are presented in Table 5. These recomputed GE values were used to compare the performance of HCSA and CSA. In literature (Chan et al. 2006), a benchmark datum was provided with six alternative paths for part manufacturing. Six new benchmark test instances (46, 47, 48, 49, 50, and 51) were generated by considering different combinations of manufacturing paths, so no result comparison existed for these six new instances because these were tested for the first time. The test instance 36 was taken from the reference (Noktehdan et al. 2016). Moreover, the test instances 38 and 44 were taken from the reference (Yang and Yang 2008) to compare our obtained HCSA results with it.

The test instances 39–42 and 45 were taken from (Ahi et al. 2009) to compare our obtained HCSA results with it.

Table 2 Description of the test instances 1 to 35

SI. nos	m	P	References	SI. nos	M	P	References
1	5	7	King and Nakornchai (1982)	19	20	23	Kumar et al. (1986)
2	5	7	Waghodekar and Sahu (1984)	20	20	35	Carrie (1973)
3	5	18	Seifoddini (1989)	21	20	35	Boe and Cheng (1991)
4	6	8	Kusiak and Cho (1992)	22	24	40	Chandrasekharan and Rajagopalan (1989)
5	7	11	Kusiak and Chow (1987)	23	24	40	Chandrasekharan and Rajagopalan (1989)
6	7	11	Boctor (1991)	24	24	40	Chandrasekharan and Rajagopalan 1989)
7	8	12	Seifoddini and Wolfe (1986)	25	24	40	Chandrasekharan and Rajagopalan 1989)
8	8	20	Chandrasekharan and Rajagopalan (1986a)	26	24	40	Chandrasekharan and Rajagopalan (1989)
9	8	20	Chandrasekharan and Rajagopalan (1986b)	27	24	40	Chandrasekharan and Rajagopalan (1989)
10	10	10	Mosier and Taube (1985a)	28	27	27	McCormick et al. (1972)
11	10	15	Chan and Milner (1982)	29	28	46	Carrie (1973)
12	14	23	Asktn and Subramantan (1987)	30	30	41	Kumar and Vannelli (1986)
13	14	24	Stanfel (1985)	31	30	50	Stanfel (1985)
14	16	24	McCormick et al. (1972)	32	30	50	Stanfel (1985)
15	16	30	Srinlvasan et al. (1990)	33	30	90	King and Nakornchai (1982)
16	16	43	King (1980)	34	37	53	McCormick et al. (1972)
17	18	24	Carrie (1973)	35	40	100	Chandrasekharan and Rajagopalan (1987)
18	20	20	Mosier and Taube (1985b)				

Table 3 First 35 test instances' recently obtained results

SI. nos	Zeb et al. (2016)	Karoum et al. (2018)	Bychkov and Batsyn (2018)	Mahmoodian et al. (2017)	(Pinheiro et al. (2017)	Batsyn et al. (2013)	(Noktehdan et al. 2016)	Shiyas and Pillai (2014)	Best GE
1	82.35	82.35	82.35	82.35	75	75	82.35	73.68	82.35
2	69.57	69.57	69.57	69.57	69.57	69.57	69.57	–	69.57
3	79.59	79.59	79.59	79.59	80.85	79.59	80.85	79.59	80.85
4	76.92	76.92	76.92	76.92	79.17	76.92	76.92	76.92	79.17
5	60.87	60.87	60.87	60.87	60.87	60.87	60.87	58.62	60.87
6	70.83	70.83	70.83	70.83	70.83	70.83	70.83	70.83	70.83
7	69.44	69.44	69.44	69.44	69.44	69.44	69.44	70.45	70.45
8	85.25	85.25	85.25	85.25	85.25	85.25	85.25	85.24	85.25
9	58.72	58.72	58.72	58.72	58.72	58.72	58.72	58.72	58.72
10	75	75	75	75	75	75	75	70.59	75
11	92	92	92	92	92	92	92	92	92
12	73.13	72.06	72.06	71.21	74.24	75	73.53	72.85	75
13	71.83	71.83	71.83	–	72.86	71.83	71.83	69.33	72.86
14	–	53.26	53.26	48.60	53.33	53.76	53.41	45.1	53.76
15	68.99	69.53	68.99	67.91	69.92	68.99	–	68.31	69.92
16	57.53	57.53	57.53	–	58.04	57.53	57.53	55.11	58.04
17	57.73	57.73	57.73	–	57.73	57.73	57.73	55.91	57.73
18	43.45	43.45	43.45	–	43.97	43.45	43.45	42.31	43.97
19	50.81	50.81	50.81	–	50.81	50.81	50.81	48.87	50.81
20	78.40	77.91	77.91	–	79.38	78.4	77.91	77.16	79.38
21	58.38	57.98	57.98	–	58.79	58.38	57.98	58.89	58.89
22	100	100	100	100	100	100	100	100	100
23	85.11	85.11	85.11	85.11	85.11	85.11	85.11	85.11	85.11
24	73.51	73.51	73.51	–	73.51	73.51	73.51	73.51	73.51
25	53.29	53.29	53.29	–	53.29	53.29	53.29	50	53.29
26	48.95	48.61	48.95	–	48.95	48.95	–	47.76	48.95
27	46.58	47.26	–	–	46.58	46.26	–	45.21	47.26
28	54.82	54.82	54.82	–	54.82	54.82	54.82	51.56	54.82
29	47.70	47.06	–	–	47.84	47.23	–	46.64	47.84
30	63.31	63.31	63.31	–	-	62.77	63.31	61.04	63.31
31	60.12	59.77	59.77	–	59.77	59.77	59.77	59.76	60.12
32	50.83	50.83	–	–	50.83	50.83	–	50.56	50.83
33	48.07	47.75	48	–	48.29	48.01	–	45.53	48.29
34	60.64	60.64	60.64	–	61.16	60.5	60.64	55.98	61.16
35	84.03	84.03	84.03	–	84.03	84.03	–	84.03	84.03

Test instance 52 was taken from a study (Imran et al. 2017) to find the process time of a part on each machine. A binary matrix was generated by considering the nonzero time values as 1's and zero time values as 0's, and hence, there was no comparison for this test instance as well. The first 35 test instances were taken from (http://mauricio.resende.info/data/cell-formation/andopt-hub.com/problems/cfp). The performance of HCSA and CSA on the first 35 test instances was compared with the results published in the literature (Noktehdan et al. 2016; Shiyas and Pillai 2014; Batsyn et al. 2013; Zeb et al. 2016; Karoum et al. 2018; Bychkov and Batsyn 2018; Mahmoodian et al. 2017; Pinheiro et al. 2017; Chaudhuri et al. 2019). The performance of HCSA and CSA over the test instances (36–45) was compared with the results presented in the references (Chan et al. 2006; Noktehdan et al. 2016; King and Nakornchai 1982; Arkat et al. 2011; Ahi et al. 2009). The performances of HCSA and CSA on the test instances from 1 to 35 and 36 to 52 are presented in Tables 5 and 6, respectively. Moreover, the statistical values of GE, such as minimum, maximum, and mean GE obtained in the 52 test instances, are also displayed in Tables 5 and 6.

Table 4 Details of the test instances from 36 to 52 with best result obtained in the literature

Sl. nos.	References	*M*	*P*	Result references	GE
36	Noktehdan et al. (2016)	8	6	Noktehdan et al. (2016)	68.75
37	Arkat et al. (2011)	10	17	Arkat et al. (2011)	50.43
38	Yang and Yang (2008)	15	15 (d)	Yang and Yang (2008)	72.58
39	Ahi et al. (2009)	12	19	Ahi et al. (2009)	46.59
40	Ahi et al. (2009)	20	20	Ahi et al. (2009)	49.5
41	Ahi et al. (2009)	25	40	Ahi et al. (2009)	36.96
42	Ahi et al. (2009)	20	51	Ahi et al. (2009)	57.87
43	Chan et al. (2006)	37	30	Chan et al. (2006)	44.07
44	Yang and Yang (2008)	28	35	Yang and Yang (2008)	70.73
45	Ahi et al. (2009)	35	18	Ahi et al. (2009)	50.71
46	Chan et al. (2006)	20	20 (1F)	–	–
47	Chan et al. (2006)	20	20 (2F)	–	–
48	Chan et al. (2006)	20	20 (3F)	–	–
49	Chan et al. (2006)	20	20 (4F)	–	–
50	Chan et al. (2006)	20	20 (5F)	–	–
51	Chan et al. (2006)	20	20 (6F)	–	–
52	Imran et al. (2017)	20	15	–	–

5.3 Statistical analysis

In the previous section, we presented an empirical comparison of results obtained by our HCSA approach based on best results obtained so far. However, this does not conclusively assert the superiority of our method since good/competitive results may sometimes be obtained by random effects and not necessarily by a proficient search.

Thus, to validate the credibility of obtained results, we have employed two statistical tests on the results achieved by the proposed HCSA for the MCFP problem, namely the Wilcoxon rank sum test and the *Z*-test. The former (Wilcoxon 1945) tests how the mean rank of optimal results differs in case of the proposed HCSA with other existing techniques over all the datasets employed for the test. The null hypothesis and alternative null hypothesis are set as follows: H_0: Median(X) is equal to Median(Y) and H_1: Median(X) is not equal to Median(Y).

The objective function for the HCSA, i.e., GE across all datasets, is tested over rest of the techniques on the specified datasets as per Tables 2 and 3. The Wilcoxon rank sum test was executed with the level of significances 0.01 and 0.05. The MATLAB function ranksum() has been used to perform the analysis. If h value is zero, then H_0 is true; otherwise, H_1 is true. For all datasets with ALPHA = 0.01 and 0.05, h value is 1 and also p value has a very small value. In this test, all the p values are less than 0.00001. This signifies that the alternative hypothesis is correct for all datasets with ALPHA = 0.01 and 0.05; or in other words, Median(X) and Median(Y) are not equal.

The *Z*-test is used to test whether the sample X is derived from a normal distribution with specified mean and standard deviation (Martinez 2007). HCSA is executed for 30 times, and we also noted the best performance at each execution. The *Z*-test is performed on group efficacy. Null hypothesis and alternative hypothesis are set as follows: H_0: $X \sim N(\mu, \sigma^2)$ and H_1: $X \nsim N(\mu, \sigma^2)$. The *Z*-test is performed by using *z*test() function available in MATLAB. The deviation between the lower and upper limits of confidence interval obtained by the *z*-test on the 52 datasets is at most 0.25. This demonstrates the significant convergence of HCSA.

5.4 Performance metrics

The important goals of multi-objective optimization are (1) finding solutions as close to the Pareto-optimal solutions as possible and (2) seeking solutions as diverse as possible in the obtained non-dominated front. In this work to test the first goal, we employ two performance metrics for the respective goals, namely generational distance (GD) for the former, and the second goal is tested by computing spacing (Deb 2001). And also to check the superiority of non-dominated solutions found using HCSA, a two-set coverage metric is used to compare HCSA with CSA.

In the metric computation, two sets are used, namely Q and P^*, where Q is the Pareto front found by test algorithm and P^* is the subset of true Pareto-optimal members. Before computing these metrics, the data in Q and P^* are to be normalized since various objective functions will have different ranges.

Table 5 Performance of HCSA and CSA on the test instances 1–35

SI. nos	Performance CSA	Performance HCSA	Minimum GE	Mean GE	After relocating the parts and machines	GAP
1	82.35	82.35	82.35	82.35	82.35	0
2	68.57	69.57	65.4	66.27	69.57	0
3	78.01	80.85	77.9	78.23	80.85	0
4	74.29	76.92	76.21	76.51	76.92	0
5	60.87	60.87	60.87	60.87	60.87	0
6	69	70.83	66.2	69.19	70.83	0
7	68.55	70.45	67	68.25	70.45	0
8	84	85.25	84	85.16	85.25	0
9	58.02	58.72	58.02	58.18	58.72	0
10	75	75	74.29	74.94	75	0
11	90.87	92	92	92	92	0
12[b]	73	75	73.17	74.46	75[b]	0
13	65	71.83	70.92	71.55	71.83	0
14***	54	54.34	54.9	55.08	53***	1.34
14[a]	51.32	52.74	51.23	51.89	52.74	0
15	66.35	69.01	68.78	68.98	68.99	0.02
16***	54.06	55.49	53.18	54.48	55.02***	0.47
16^{+++}	54.06	55.49	53.18	54.48	55.02^{+++}	0.47
17	55.5	57.73	56	57.14	57.73	0
18	43.11	43.45	42.29	43.1	43.45	0
19	48.8	50.81	49.1	50.26	50.81	0
20***	75.21	76.7	75.16	75.98	76.47***	0.23
20^{+++}	74.21	75.43	74.1	75.24	75.43^{+++}	0
21***	55.32	58.06	57.03	57.4	58.06***	0
21^{+++}	56.17	57.21	55.43	56.83	57.21^{+++}	0
22	96.63	100	97	99.31	100	0
23	84.29	85.11	83	84.28	85.11	0
24	73.1	73.51	70	72.47	73.51	0
25	52.12	53.29	51	52.93	53.29	0
26	47.71	48.95	47.17	48.66	48.95	0
27	45.93	46.26	45.81	46.07	46.26	0
28	53.37	54.82	54.03	54.64	54.82	0
29***	39.18	40.18	37	39.36	39.29***	0.89
30	60.28	63.31	60.9	62.42	63.31	0
31	58.82	59.77	58.18	59.49	59.77	0
32	47.79	50.83	49.21	50.55	50.83	0
33***	41.43	45.32	44	45.02	42.67***	2.65
34***	53.69	54.73	53.91	54.6	31.78***	22.95
34^{+++}	53.69	54.73	53.91	54.6	31.78^{+++}	22.95
35	80.1	84.03	79	82.01	84.03	0

'***' and '+++' in the table indicate that these GE values have been obtained after relocation of the machines and parts into appropriate cells from (Batsyn et al. 2013; Bychkov and Batsyn 2018), respectively, 'b' represents that GE value is taken from (Batsyn et al. 2013), and 'a' represents removal of zero-sum rows and columns

Table 6 Performance of HCSA and CSA on test instances 36–52

SI. nos.	Maximum GE obtained by HCSA	Minimum GE obtained by HCSA	Average GE obtained by HCSA	Maximum GE obtained by CSA	Base paper GE	Improvement in GE By HCSA
36	68.75	66.21	67.34	67.85	68.75	0
37	50.43	42	48.27	49.12	50.43	0
38	72.58	70.16	71.18	71.18	72.58	0
39	55.45	54.32	55.26	54.02	46.59	8.86
40	69.04	66	68.84	68.21	49.5	19.54
41	62.5	59.37	61.63	61.36	36.96	25.54
42	60.34	57	59.94	59.2	57.87	2.47
43	53.79	52	53.16	52.01	44.07	9.72
44	70.73	62	64.73	65.42	70.73	0
45	66.25	64	65.59	63.6	50.71	15.54
46	72.3	70.11	71.16	72	–	–
47	71.42	69.7	71.21	70.93	–	–
48	71.05	70.67	70.94	71.05	–	–
49	70.66	68.42	69.81	69.54	–	–
50	69.23	68.21	68.88	68.8	–	–
51	68.29	67.41	68.02	67.07	–	–
52	45.6	42	43.89	43.16	–	–

5.5 Generational distance

Veldhuizen introduces this metric in the year 1990 (Van Veldhuizen and David 1999). This metric finds an average distance between the members of Q and P^*. An algorithm having a small value of GD is better. In this research paper, the criteria followed for the construction of approximate optimal front P^* are as follows: Find the minimum value of each objective function from the set of objective function vectors in the non-dominated set. The set of all the vectors in objective space from the non-dominated set with at least one minimum value exists, and then, it belongs to the optimal front.

5.6 Spacing (SP)

Schott introduces this metric in the year 1995 (Schott 1995). A good algorithm will be having minimal SP value. The set Q is caught by executing HCSA for 50 iterations with each iteration having 50 agents. In all iterations, the Pareto fronts are stored in external memory (DB-1). The metrics for GD and SP for all the three objectives (voids, exceptional elements, and inter-cell movements) and for the algorithms HCSA and CSA are provided in Table 7.

To verify the superiority of HCSA over CSA in the multiple objectives, two-set coverage metric is computed and the details are given as follows:

5.7 Set coverage metric

To understand the relative spread of solutions of CSA and HCSA, set coverage metric is used which is proposed by Zitzler in the year 1999 (Zitzler 1999). The set coverage metric values associated with HCSA and CSA, and CSA and HCSA are given in Table 8. In CSA, initial population is generated randomly without following any cell sequence. From Table 8, C(HCSA, CSA) values are superior than C(CSA, HCSA) for majority of the problem instances. And also the total number of non-dominated solutions obtained by HCSA and CSA on the 52 test instances is given in Table 9.

5.8 Scalability issues of HCSA

For the small test instances from 1 (5×7) to 10 (10×10), HCSA has reached global optimum within 13 iterations. For moderate size instances from 11 to 28, HCSA has reached global optimum within 29 iterations, whereas for the large instances from 29 to 35, HCSA took 47 iterations to reach global optimum. For the test instances 36–52, HCSA took on average 26 iterations to reach best value for GE. Hence, the total number of iterations is fixed to 50 in HCSA.

Table 7 GD and SP metric values for all datasets Pareto fronts

Sl. nos.	GD	SP	Sl. nos.	GD	SP	SI. nos	GD	SP	Sl. nos.	GD	SP
Hybrid clonal selection algorithm											
1	0.38	0.92	14	0.21	0.29	27	0.17	0.61	40	0.40	0.58
2	0.27	1.07	15	0.14	0.75	28	0.21	0.48	41	0.12	0.72
3	0.15	0.87	16	0.31	0.27	29	0.18	0.61	42	0.34	0.29
4	0.49	1.3	17	0.21	0.44	30	0.31	1.01	43	0.31	0.22
5	0.63	1.01	18	0.15	0.31	31	0.42	0.92	44	0.56	0.18
6	0.12	0.97	19	0.14	0.42	32	0.42	0.36	45	0.52	0.26
7	0.29	0.47	20	0.20	0.71	33	0.12	0.78	46	0.49	0.37
8	0.21	0.25	21	0.23	0.31	34	0.13	0.21	47	0.27	0.12
9	0.11	0.14	22	0.67	0.12	35	0.22	0.12	48	0.28	0.71
10	0.41	0.71	23	0.81	0.42	36	0.42	0.48	49	0.23	0.42
11	0.22	0.87	24	0.50	0.72	37	0.23	0.39	50	0.24	0.39
12	0.13	0.6	25	0.38	0.45	38	0.38	0.55	51	0.27	0.33
13	0.28	0.37	26	0.29	0.43	39	0.13	0.62	52	0.81	0.36
Clonal selection algorithm											
1	0.17	0.61	14	0.32	0.17	27	0.12	0.23	40	0.41	0.14
2	0.16	0.81	15	0.11	0.18	28	0.17	0.11	41	0.12	0.28
3	0.18	0.61	16	0.29	0.21	29	0.38	0.51	42	0.13	0.21
4	0.31	1.01	17	0.13	0.16	30	0.23	0.69	43	0.12	0.12
5	0.42	0.92	18	0.10	0.17	31	0.45	0.41	44	0.52	0.31
6	0.11	0.34	19	0.20	0.26	32	0.62	0.40	45	0.25	0.16
7	0.12	0.78	20	0.11	0.59	33	0.52	0.71	46	0.42	0.17
8	0.13	0.21	21	0.17	0.24	34	0.31	0.31	47	0.32	0.11
9	0.12	0.12	22	0.29	0.28	35	0.25	0.16	48	0.19	0.87
10	0.41	0.41	23	0.43	0.31	36	0.41	0.41	49	0.12	0.81
11	0.23	0.31	24	0.22	0.52	37	0.32	0.16	50	0.64	0.27
12	0.21	0.6	25	0.44	0.49	38	0.24	0.21	51	0.27	0.71
13	0.34	0.37	26	0.11	0.51	39	0.40	0.71	52	0.72	0.46

Table 8 Two-set coverage metric values for HCSA and CSA approaches

Instance number	*C* (HCSA, CSA)	*C* (CSA, HCSA)	Instance number	C (HCSA, CSA)	C (CSA, HCSA)	Instance number	C (HCSA, CSA)	C(CSA, HCSA)	Instance number	C(HCSA, CSA)	C(CSA, HCSA)
1	0.721	0.156	14	0.814	0.319	27	0.239	0.406	40	0.551	0.314
2	0.822	0.242	15	0.937	0.302	28	0.81	0.181	41	0.428	0.517
3	0.633	0.437	16	0.559	0.113	29	0.488	0.524	42	0.681	0.719
4	0.715	0.411	17	0.501	0.153	30	0.502	0.201	43	0.581	0.712
5	0.552	0.21	18	0.498	0.118	31	0.726	0.492	44	0.632	0.219
6	0.781	0.113	19	0.831	0.227	32	0.621	0.82	45	0.32	0.61
7	0.919	0.419	20	0.607	0.199	33	0.416	0.291	46	0.791	0.81
8	0.614	0.276	21	0.422	0.21	34	0.36	0.25	47	0.4	0.51
9	0.927	0.117	22	0.72	0.61	35	0.81	0.34	48	0.312	0.29
10	0.59	0.218	23	0.21	0.34	36	0.367	0.352	49	0.681	0.41
11	0.914	0.101	24	0.613	0.759	37	0.75	0.56	50	0.182	0.27
12	0.771	0.219	25	0.492	0.51	38	0.214	0.391	51	0.532	0.416
13	0.611	0.117	26	0.625	0.583	39	0.861	0.452	52	0.761	0.819

Table 9 Total number of non-dominated solutions found for each problem instance by HCSA and CSA

Problem instances	HCSA	CSA	Problem instances	HCSA	CSA	Problem instances	HCSA	CSA	Problem instance	HCSA	CSA
1	66	31	14	27	25	27	25	31	40	76	16
2	49	28	15	20	20	28	92	18	41	42	10
3	57	40	16	35	21	29	71	40	42	50	19
4	30	20	17	48	52	30	5	10	43	81	26
5	32	34	18	33	19	31	22	41	44	21	9
6	13	11	19	36	37	32	31	11	45	13	24
7	24	26	20	55	35	33	43	21	46	36	15
8	48	50	21	27	10	34	48	50	47	17	12
9	20	15	22	19	5	35	30	25	48	12	11
10	43	38	23	32	25	36	35	21	49	20	12
11	36	30	24	25	13	37	53	12	50	29	32
12	12	28	25	31	21	38	42	31	51	42	11
13	55	22	26	30	14	39	29	12	52	18	21

5.9 Discussion

This section summarizes the overall quantitative outcomes of subsections 5.2, 5.3, and 5.4. It can be observed from the results that the convergence and superiority of HCSA in terms of GE and also all three objectives of the proposed MCFP are evident. It was also found that for 27% problems, HCSA showed improved grouping efficacy, and the sum of obtained GEs was 130.76. For remaining cases, the proposed HCSA yielded near-optimal solutions. In terms of group efficacy, the deviation of obtained near-optimal solutions varied between only 1.26 and 4.34%. On the contrary, it can be noticed from Table 6 that for particular datasets, the performance gain reached 25.54%. The average value of the main three objectives (voids, exceptional elements, and inter-cell movements) for all 52 datasets was 19.83, 28.61, and 24.72, respectively. Moreover, for smaller problem instances, HCSA displayed almost comparable results. It is worth noting that for certain significant cases, performance improvements were considerably high. Thus, it can be inferred that the proposed HCSA is a superior solution for MCFP. Finally, to validate the superiority of HCSA, two tests were carried out, at first, the Wilcoxon rank sum tests for statistical testing and then *Z*-test to reveal the convergence of the solutions produced by HCSA.

6 Conclusion

The multi-objective optimization of the machine cell part family problem is a proved to be an NP-hard problem. In this paper, we presented a hybrid clonal selection algorithm for this problem that employs genetic algorithm to generate feasible cell sequences within clonal selection part itself. Since all solutions produced by the proposed HCSA are feasible solutions, there is no need to check any constraints as required by the problem formulation. Thus, the proposed HCSA need not convert infeasible solutions and thus is performed efficiently as expected. The proposed algorithm has been tested with 52 numerical datasets (various benchmark datasets' size varying from 35 to 4000) drawn from existing literature. The comparative study of fourteen competent algorithms on the same datasets clearly shows the efficiency of proposed algorithm in terms of group efficacy. Moreover, based on the obtained results, it can be observed that HCSA shows near-optimal or better when compared to existing MCFP solutions using other techniques. Furthermore, extensive statistical tests and convergence tests have been carried out to ascertain its superiority of the proposed HCSA. Hence, we conclude that the proposed HCSA is quite competitive with respective to the current state of the art. While working on this, we observed the similarities in rows and columns in the datasets which have triggered our interest to construct neighborhood hyperedges for building hypergraph structures with known topological and geometrical properties that can be exploited for obtaining elegant and quick solutions for the MCFP with large-sized data.

Acknowledgement Madhusudana Rao Nalluri and K. Kannan gratefully acknowledge Tata Realty-IT city-SASTRA Srinivasa Ramanujan Research Cell of our university (India) for the financial support extended to us in carrying out this research work. Xiao-Zhi Gao's research work was partially supported by the National Natural Science Foundation of China (NSFC) under Grant 51875113.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

Abdel-Basset M, Fakhry AE, El-Henawy I, Qiu T, Sangaiah AK (2017a) Feature and intensity based medical image registration using particle swarm optimization. J Med Syst 41(12):197

Abdel-Basset M, El-Shahat D, Sangaiah AK (2017b) A modified nature inspired meta-heuristic whale optimization algorithm for solving 0–1 knapsack problem. Int J Mach Learn Cybern 1:1–20

Abdel-Basset M, El-Shahat D, El-Henawy I, Sangaiah AK (2018a) A modified flower pollination algorithm for the multidimensional knapsack problem: human-centric decision making. Soft Comput 22(13):4221–4239

Abdel-Basset M, El-Shahat D, El-henawy I, Sangaiah AK, Ahmed SH (2018b) A novel whale optimization algorithm for cryptanalysis in Merkle–Hellman cryptosystem. Mob Netw Appl 5:1–11

Ahi A, Aryanezhad MB, Ashtiani B, Makui A (2009) A novel approach to determine cell formation, intracellular machine layout and cell layout in the CMS problem based on TOPSIS method. Comput Oper Res 36(5):1478–1496

Albadawi Z, Bashir HA, Chen M (2005) A mathematical approach for the formation of manufacturing cells. Comput Ind Eng 48(1):3–21

Arikan F, Güngör Z (2005) A parametric model for cell formation and exceptional elements' problems with fuzzy parameters. J Intell Manuf 16(1):103–114

Arkat J, Hosseini L, Farahani MH (2011) Minimization of exceptional elements and voids in the cell formation problem using a multi-objective genetic algorithm. Expert Syst Appl 38(8):9597–9602

Arkat J, Abdollahzadeh H, Ghahve H (2012) A new branch and bound algorithm for cell formation problem. Appl Math Model 36(10):5091–5100

Asktn RG, Subramantan SP (1987) A cost-based heuristic for group technology configuration. Int J Prod Res 25(1):101–113

Bajestani MA, Rabbani M, Rahimi-Vahed AR, Khoshkhou GB (2009) A multi-objective scatter search for a dynamic cell formation problem. Comput Oper Res 36(3):777–794

Balakrishnan SM, Sangaiah AK (2017a) MIFIM—middleware solution for service centric anomaly in future internet models. Future Gener Comput Syst 74:349–365

Balakrishnan SM, Sangaiah AK (2017b) Integrated QoUE and QoS approach for optimal service composition selection in internet of services (IoS). Multimed Tools Appl 76(21):22889–22916

Batsyn M, Bychkov I, Goldengorin B, Pardalos P, Sukhov P (2013) Pattern-based heuristic for the cell formation problem in group technology. In: Goldengorin B (ed) Models, algorithms, and technologies for network analysis, pp 11–50. Springer, New York

Boctor FF (1991) A linear formulation of the machine-part cell formation problem. Int J Prod Res 29(2):343–356

Boe WJ, Cheng CH (1991) A close neighbour algorithm for designing cellular manufacturing systems. Int J Prod Res 29(10):2097–2116

Burbidge JL (1971) Production flow analysis. Prod Eng 50(4.5):139–152

Bychkov I, Batsyn M (2018) An efficient exact model for the cell formation problem with a variable number of production cells. Comput Oper Res 91:112–120

Cai X, Gao XZ, Xue Y (2016) Improved bat algorithm with optimal forage strategy and random disturbance strategy. Int J Bio-Inspir Comput 8(4):205–214

Cai X, Wang H, Cui Z, Cai J, Xue Y, Wang L (2018) Bat algorithm with triangle-flipping strategy for numerical optimization. Int J Mach Learn Cybern 9(2):199–215

Carrie AS (1973) Numerical taxonomy applied to group technology and plant layout. Int J Prod Res 11(4):399–416

Chan HM, Milner DA (1982) Direct clustering algorithm for group formation in cellular manufacture. J Manuf Syst 1(1):65–75

Chan FT, Lau KW, Chan PL, Choy KL (2006) Two-stage approach for machine-part grouping and cell layout problems. Robot Comput Integr Manuf 22(3):217–238

Chandrasekharan M, Rajagopalan R (1986a) MODROC: an extension of rank order clustering for group technology. Int J Prod Res 24(5):1221–1233

Chandrasekharan PM, Rajagopalan R (1986b) An ideal seed non-hierarchical clustering algorithm for cellular manufacturing. Int J Prod Res 24(2):451–463

Chandrasekharan MP, Rajagopalan R (1987) ZODIAC—an algorithm for concurrent formation of part-families and machine-cells. Int J Prod Res 25(6):835–850

Chandrasekharan MP, Rajagopalan R (1989) Groupabil1ty: an analysis of the properties of binary data matrices for group technology. Int J Prod Res 27(6):1035–1052

Chattopadhyay M, Dan PK, Mazumdar S (2012) Application of visual clustering properties of self organizing map in machine-part cell formation. Appl Soft Comput 12(2):600–610

Chaudhuri B, Jana RK, Sharma DK, Dan PK (2019) A goal programming embedded genetic algorithm for multi-objective manufacturing cell design. Int J Appl Decis Sci 12(1):98–114

Chen SJ, Cheng CS (1995) A neural network-based cell formation algorithm in cellular manufacturing. Int J Prod Res 33(2):293–318

Chiang HS, Sangaiah AK, Chen MY, Liu JY (2018) A novel artificial bee colony optimization algorithm with SVM for bio-inspired software-defined networking. Int J Parallel Program 1:1–19

Chu CH (1997) An improved neural network for manufacturing cell formation. Decis Support Syst 20(4):279–295

Cui Z, Sun B, Wang G, Xue Y, Chen J (2017) A novel oriented cuckoo search algorithm to improve DV-hop performance for cyber–physical systems. J Parallel Distrib Comput 103:42–52

Cui Z, Cao Y, Cai X, Cai J, Chen J (2018) Optimal LEACH protocol with modified bat algorithm for big data sensing systems in Internet of Things. J Parallel Distrib Comput. https://doi.org/10.1016/j.jpdc.2017.12.014

Dalfard VM (2013) New mathematical model for problem of dynamic cell formation based on number and average length of intra and intercellular movements. Appl Math Model 37(4):1884–1896

De Castro LN, Von Zuben FJ (2002) Learning and optimization using the clonal selection principle. IEEE Trans Evolut Comput 6(3):239–251

Deb K (2001) Multi-objective optimization using evolutionary algorithms, vol 16. Wiley, New York

Delgoshaei A, Gomes C (2016) A multi-layer perceptron for scheduling cellular manufacturing systems in the presence of unreliable machines and uncertain cost. Appl Soft Comput 49:27–55

Dimopoulos C, Zalzala AM (2000) Recent developments in evolutionary computation for manufacturing optimization: problems, solutions, and comparisons. IEEE Trans Evolut Comput 4(2):93–113

Floreano D, Mattiussi C (2008) Bio-inspired artificial intelligence: theories, methods, and technologies. MIT Press, London

Gonçalves JF, Resende MG (2004) An evolutionary algorithm for manufacturing cell formation. Comput Ind Eng 47(2):247–273

Guerrero F, Lozano S, Smith KA, Canca D, Kwok T (2002) Manufacturing cell formation using a new self-organizing neural network. Comput Ind Eng 42(2):377–382

Ham I, Hitomi K, Yoshida T (1985) Group technology: applications to production management. Kluwer Academic Publications, Alphen aan den Rijn

Hwang H, Sun JU (1996) A genetic-algorithm-based heuristic for the GT cell formation problem. Comput Ind Eng 30(4):941–955

Imran M, Kang C, Lee YH, Jahanzaib M, Aziz H (2017) Cell formation in a cellular manufacturing system using simulation integrated hybrid genetic algorithm. Comput Ind Eng 105:123–135

James TL, Brown EC, Keeling KB (2007) A hybrid grouping genetic algorithm for the cell formation problem. Comput Oper Res 34(7):2059–2079

Karoum B, Elbenani B, El Khattabi N, El Imrani AA (2018) Manufacturing cell formation problem using hybrid Cuckoo search algorithm. In: Amodeo L (ed) Recent developments in metaheuristics, pp 151–162. Springer, Cham

Kim J, Bentley PJ (2001) Towards an artificial immune system for network intrusion detection: an investigation of clonal selection with a negative selection operator. In: Proceedings of the 2001 congress on evolutionary computation, IEEE, vol 2, pp 1244–1252

King JR (1980) Machine-component grouping in production flow analysis: an approach using a rank order clustering algorithm. Int J Prod Res 18(2):213–232

King JR, Nakornchai V (1982) Machine-component group formation in group technology: review and extension. Int J Prod Res 20(2):117–133

Kumar KR, Vannelli A (1986) Strategic subcontracting for efficient disaggregated manufacturing. BEBR faculty working paper; no. 1252

Kumar KR, Kusiak A, Vannelli A (1986) Grouping of parts and components in flexible manufacturing systems. Eur J Oper Res 24(3):387–397

Kusiak A, Cho M (1992) Similarity coefficient algorithms for solving the group technology problem. Int J Prod Res 30(11):2633–2646

Kusiak A, Chow WS (1987) Efficient solving of the group technology problem. J Manuf Syst 6(2):117–124

Li X, Baki MF, Aneja YP (2010) An ant colony optimization metaheuristic for machine-part cell formation problems. Comput Oper Res 37(12):2071–2081

Lin TL, Dessouky MM, Kumar KR, Ng SM (1996) A heuristic-based procedure for the weighted production-cell formation problem. IIE Trans 28(7):579–590

Lu X, Tang K, Sendhoff B, Yao X (2014) A review of concurrent optimisation methods. Int J Bio-Inspir Comput 6(1):22–31

Mahdavi I, Paydar MM, Solimanpur M, Heidarzade A (2009) Genetic algorithm approach for solving a cell formation problem in cellular manufacturing. Expert Syst Appl 36(3):6598–6604

Mahmoodian V, Jabbarzadeh A, Rezazadeh H, Barzinpour F (2017) A novel intelligent particle swarm optimization algorithm for solving cell formation problem. Neural Comput Appl 1:1–15

Martinez WL, Martinez AR (2007) Computational statistics handbook with MATLAB, vol 22. CRC Press, London

McAuley J (1972) Machine grouping for efficient production. Prod Eng 51:53–57

McCormick WT Jr, Schweitzer PJ, White TW (1972) Problem decomposition and data reorganization by a clustering technique. Oper Res 20(5):993–1009

Mitranov SP (1959) The scientific principles of group technology. National Lending Library, London

Mosier C, Taube L (1985a) The facets of group technology and their impacts on implementation—a state-of-the-art survey. Omega 13(5):381–391

Mosier C, Taube L (1985b) Weighted similarity measure heuristics for the group technology machine clustering problem. Omega 13(6):577–579

Nalluri MR, Roy DS (2017) Hybrid disease diagnosis using multiobjective optimization with evolutionary parameter optimization. J Healthc Eng 2017:27

Nedjah N, Mourelle LDM (2015) Evolutionary multi-objective optimisation: a survey. Int J Bio-Inspir Comput 7(1):1–25

Noktehdan A, Seyedhosseini S, Saidi-Mehrabad M (2016) A Metaheuristic algorithm for the manufacturing cell formation problem based on grouping efficacy. Int J Adv Manuf Technol 82(1–4):25–37

Nouri H (2016) Development of a comprehensive model and BFO algorithm for a dynamic cellular manufacturing system. Appl Math Model 40(2):1514–1531

Oliveira S, Ribeiro JFF, Seok SC (2008) A comparative study of similarity measures for manufacturing cell formation. J Manuf Syst 27(1):19–25

Oliveira S, Ribeiro JFF, Seok SC (2009) A spectral clustering algorithm for manufacturing cell formation. Comput Ind Eng 57(3):1008–1014

Onwubolu GC, Mutingi M (2001) A genetic algorithm approach to cellular manufacturing systems. Comput Ind Eng 39(1):125–144

Pailla A, Trindade AR, Parada V, Ochi LS (2010) A numerical comparison between simulated annealing and evolutionary approaches to the cell formation problem. Expert Syst Appl 37(7):5476–5483

Pandian RS, Mahapatra SS (2009) Manufacturing cell formation with production data using neural networks. Comput Indus Eng 56(4):1340–1347

Panigrahi BK, Yadav SR, Agrawal S, Tiwari MK (2007) A clonal algorithm to solve economic load dispatch. Electr Power Syst Res 77(10):1381–1389

Papaioannou G, Wilson JM (2010) The evolution of cell formation problem methodologies based on recent studies (1997–2008): review and directions for future research. Eur J Oper Res 206(3):509–521

Pinheiro RGS, Martins IC, Protti F, Ochi LS (2017) A matheuristic for the cell formation problem. Opt Lett 12:1–12

Rajesh KD, Krishna MM, Ali MA, Chalapathi PV (2017) A modified hybrid similarity coefficient based method for solving the cell formation problem in cellular manufacturing system. Mater Today Proc 4(2):1469–1477

Rao NM, Kannan K, Gao XZ, Roy DS (2018) Novel classifiers for intelligent disease diagnosis with multi-objective parameter evolution. Comput Electr Eng 67:483–496

Saeidi S, Solimanpur M, Mahdavi I, Javadian N (2014) A multiobjective genetic algorithm for solving cell formation problem using a fuzzy goal programming approach. Int J Adv Manuf Technol 70(9–12):1635–1652

Safaei N, Saidi-Mehrabad M, Tavakkoli-Moghaddam R, Sassani F (2008) A fuzzy programming approach for a cell formation problem with dynamic and uncertain conditions. Fuzzy Sets Syst 159(2):215–236

Sahin YB, Alpay S (2016) A metaheuristic approach for a cubic cell formation problem. Expert Syst Appl 65:40–51

Sakhaii M, Tavakkoli-Moghaddam R, Bagheri M, Vatani B (2016) A robust optimization approach for an integrated dynamic cellular manufacturing system and production planning with unreliable machines. Appl Math Model 40(1):169–191

Schott JR (1995) Fault tolerant design using single and multicriteria genetic algorithm optimization. No. AFIT/CI/CIA-95-039. Air force inst of tech Wright–Patterson AFB OH

Seifoddini HK (1989) Single linkage versus average linkage clustering in machine cells formation applications. Comput Ind Eng 16(3):419–426

Seifoddini H, Wolfe PM (1986) Application of the similarity coefficient method in group technology. IIE Trans 18(3):271–277

Selim HM, Askin RG, Vakharia AJ (1998) Cell formation in group technology: review, evaluation and directions for future research. Comput Ind Eng 34(1):3–20

Selim HM, Abdelaal RM, Mahdi AI (2003) Formation of machine groups and part families: a modified SLC method and comparative study. Integr Manuf Syst 14(2):123–137

Shiyas CR, Pillai VM (2014) A mathematical programming model for manufacturing cell formation to develop multiple configurations. J Manuf Syst 33(1):149–158

Solimanpur M, Vrat P, Shankar R (2004) A multi-objective genetic algorithm approach to the design of cellular manufacturing systems. Int J Prod Res 42(7):1419–1441

Srikanth K, Panwar LK, Panigrahi BK, Herrera-Viedma E, Sangaiah AK, Wang GG (2018) Meta-heuristic framework: quantum inspired binary grey wolf optimizer for unit commitment problem. Comput Electr Eng 70:243–260

Srinlvasan G, Narendran TT, Mahadevan B (1990) An assignment model for the part-families problem in group technology. Int J Prod Res 28(1):145–152

Stanfel LE (1985) Machine clustering for economic production. Eng Costs Prod Econ 9(1–3):73–81

Su CT, Hsu CM (1998) Multi-objective machine-part cell formation through parallel simulated annealing. Int J Prod Res 36(8):2185–2207

Suresh Kumar C, Chandrasekharan MP (1990) Grouping efficacy: a quantitative criterion for goodness of block diagonal forms of binary matrices in group technology. Int J Prod Res 28(2):233–243

Tsai CC, Lee CY (2006) Optimization of manufacturing cell formation with a multi-functional mathematical programming model. Int J Adv Manuf Technol 30(3–4):309–318

Van Veldhuizen, DA (1999). Multiobjective evolutionary algorithms: classifications, analyses, and new innovations. No. AFIT/DS/ENG/99-01. Air Force Institute of Technology Wright–Patterson AFB Oh School of Engineering

Verma A, Kaushal S, Sangaiah AK (2017) Computational intelligence based heuristic approach for maximizing energy efficiency in internet of things. In: Sangaiah AK (ed) Intelligent decision support systems for sustainable computing, pp 53–76. Springer, Cham

Waghodekar PH, Sahu S (1984) Machine-component cell formation in group technology: MACE. Int J Prod Res 22(6):937–948

Wemmerlöv U, Hyer NL (1987) Research issues in cellular manufacturing. Int J Prod Res 25(3):413–431

Wemmerlov U, Johnson DJ (1997) Cellular manufacturing at 46 user plants: implementation experiences and performance improvements. Int J Prod Res 35(1):29–49

White JA, Garrett SM (2003) Improved pattern recognition with artificial clonal selection? In: Artificial immune systems, pp 181–193. Springer, Berlin

Wilcoxon Frank (1945) Individual comparisons by ranking methods. Biom Bull 1(6):80–83

Wu TH, Chung SH, Chang CC (2010) A water flow-like algorithm for manufacturing cell formation problems. Eur J Oper Res 205(2):346–360

Yang MS, Yang JH (2008) Machine-part cell formation in group technology using a modified ART1 method. Eur J Oper Res 188(1):140–152

Yang JH, Sun L, Lee HP, Qian Y, Liang YC (2008) Clonal selection based memetic algorithm for job shop scheduling problems. J Bionic Eng 5(2):111–119

Yin Y, Yasuda K (2005) Similarity coefficient methods applied to the cell formation problem: a comparative investigation. Comput Ind Eng 48(3):471–489

Zeb A, Khan M, Khan N, Tariq A, Ali L, Azam F, Jaffery SHI (2016) Hybridization of simulated annealing with genetic algorithm for cell formation problem. Int J Adv Manuf Technol 86(5–8):2243–2254

Zhang M, Wang H, Cui Z, Chen J (2018) Hybrid multi-objective cuckoo search with dynamical local search. Memet Comput 10(2):199–208

Zheng Z, Saxena N, Mishra KK, Sangaiah AK (2018) Guided dynamic particle swarm optimization for optimizing digital image watermarking in industry applications. Future Gener Comput Syst 5:256

Zitzler E (1999) Evolutionary algorithms for multiobjective optimization: methods and applications. Ph.D. thesis, ETH Zurich, Switzerland