Deploying NSBA algorithm for Bi-Objective Manufacturing Cells Considering Percentage Utilization of Machines

Tamal Ghosh[1]* and Kristian Martinsen[2]

[1, 2] Department of Manufacturing and Civil Engineering,
Norwegian University of Science and Technology,
Teknologivegen 22, 2815 Gjøvik, Norway
Email: tamal.ghosh@ntnu.no [1]
Email: kristian.martinsen@ntnu.no [2]
Phone: +47-735-91-781
*Corresponding Author

Abstract: Percentage Utilization of Machines is considered as an important production factor for manufacturing Cell Formation Problem (CFP) in Cellular Manufacturing (CM). This recently developed concept correctly emphasize ration data in context of CM. In this paper, a utilization based bi-objective mathematical model is developed, which minimizes the total machine utilization induced by bottleneck machines and number of voids. Thereafter, a new data generating algorithm is introduced. The abovementioned bi-objective CFP is solved using a Non-Dominated Sorting Bat Algorithm (NSBA), which is compared with published Multi-Objective Bat Algorithm (MOBA) successfully. Statistical tests are conducted and data consistency is confirmed on obtained results. The computational experiments depict that the Pareto solutions of NSBA are 35.7% improved. The contribution of this research is threefold. First, an accurate bi-objective mathematical expression is developed for utilization based CFPs. Second, a novel data generating algorithm is stated. Third, NSBA technique is successfully tested.

Keywords: Machine Utilization Percentage; Cellular Manufacturing; NSBA; Bi-Objective Mathematical Model

Biographical notes:

Tamal Ghosh is a Postdoctoral Researcher at the Department of Manufacturing and Civil Engineering, Norwegian University of Science and Technology in Gjøvik. He has obtained his PhD from the Production Engineering Department, Jadavpur University, India. Prior to this, he received his Master of Technology (M.Tech) in Industrial Engineering from West Bengal University of Technology and Bachelor of Technology (B.Tech) in Computer Science and Engineering from National Institute of Technology Calicut, India. He has published articles in international journals and conferences. His current research interests include bio-inspired and evolutionary computing, deep learning, artificial intelligence, mathematical modelling, and simulations.

Kristian Martinsen is a Professor and manager for the research group of sustainable manufacturing at the Department of Manufacturing and civil Engineering, Norwegian University of Science and Technology in Gjøvik. Prior to this, he has held various research and management positions at Raufoss ASA and Raufoss Technology and Industrial Management AS /SINTEF Raufoss Manufacturing AS for around 20 years. He has obtained his Masters and PhD from NTNU in during 1989-1995. He has been a successful editor for various projects of IPN, SFI and EU. He has published research papers in various international journals and conferences. He has guided several PhDs. His current research interests include manufacturing engineering, sustainable manufacturing, manufacturing process monitoring and control, metal cutting, additive manufacturing, measuring technology, tolerance engineering, industry 4.0, and learning factories.

Deploying NSBA algorithm for Bi-Objective Manufacturing Cells Considering Percentage Utilization of Machines

Abstract: Percentage Utilization of Machines is considered as an important production factor for manufacturing Cell Formation Problem (CFP) in Cellular Manufacturing (CM). This recently developed concept correctly emphasize ration data in context of CM. In this paper, an utilization based bi-objective mathematical model is developed, which minimizes the total machine utilization induced by bottleneck machines and number of voids. Thereafter, a new data generating algorithm is introduced. The abovementioned bi-objective CFP is solved using a Non-Dominated Sorting Bat Algorithm (NSBA), which is compared with published Multi-Objective Bat Algorithm (MOBA) successfully. Statistical tests are conducted and data consistency is confirmed on obtained results. The computational experiments depict that the Pareto solutions of NSBA are 35.7% improved. The contribution of this research is threefold. First, an accurate bi-objective mathematical expression is developed for utilization based CFPs. Second, a novel data generating algorithm is stated. Third, NSBA technique is successfully tested.

Keywords: Machine Utilization Percentage; Cellular Manufacturing; NSBA; Bi-Objective Mathematical Model

1 Introduction

Group Technology (GT) and its application of Cellular Manufacturing (CM) play a significant role in manufacturing industry since decades (Goldengorin, et al., 2013). Tactically GT forms part families based on similarities in attributes or processing requirements and assigns them to the suitable machine groups to make the most of the mass production in terms of enhanced throughput times, minimized work in process, reduction in tool requirements, product quality enhancement and improvement in overall control of operations (Chattopadhyay, et al., 2014). The main objective of CM is to decompose the production system into several miniature systems which practically exploit the processing resemblances of parts and form groups of dissimilar machines (Selim, et al., 1998). Being a function of GT, CM presents a combined setup of jobshop (product mix) and flowshop (higher rate of production) which demonstrates an efficient alternative of traditional production system.

The procedure of allocating part families to the machine cells, is termed as the production Cell Formation Problem (CFP) in the vicinity of Cellular Manufacturing. CFP makes use of the classical Machine-Part Incidence Matrix (MPIM) and attains block diagonal cellular structure to generate cells. An MPIM is bundled with '0' and '1' depending upon the machining requirements of parts. It is popularly known as 'binary data'. If the '1' elements are replaced with processing time of parts, it is termed as ratio data or workload data (Sengupta, et al., 2011). Binary data are mostly used in the past researches in CM (Mccormick, et al., 1972); (King, 1980); (Chandrasekaran & Rajagopalan, 1986); (Srinivasan, 1994); (Dimopoulos & Mort, 2001); (Xambre & Vilarinho, 2003); (Durán, et al., 2008); (Sayadi, et al., 2013); (Žilinskas, et al., 2014).

Workload data were first explained by ref. (Venugopal & Narendran, 1992). Thereafter some researchers have incorporated ratio data to solve CFP (George, et al., 2003); (Mahapatra & Pandian, 2008). The total processing time on a machine/work station for any part could be obtained by multiplying its production volume and unit processing time. All the '1's in the MPIM are then converted to workload values. These would take any value in the ratio scale (0-1) and termed as the ratio level data (Sengupta, et al., 2011). However past researches actually failed to show the systematic way to obtain the workload value from total processing time. Recently (Ghosh, et al., 2017) has demonstrated a technique to portray workload data realistically.

The production cell design problem is NP-hard in nature (Dimopoulos & Zalzala, 2000). Thus a lot of attention have been offered while developing suitable methodologies to obtain optimal solutions for the stated problem. In recent past, a number of review articles have appeared based on solution methodologies and techniques (Papaioannou & Wilson, 2010); (Arora, et al., 2013); (Chattopadhyay, et al., 2013). These methodologies can be primarily categorized as mathematical programming based approaches, bio-inspired techniques such as neural networks and meta-heuristics algorithms (Papaioannou & Wilson, 2010); (Ghosh, et al., 2011). These are exclusively Genetic Algorithms (GA) (Gupta, et al., 1996); (Zolfaghari & Liang, 2003); (Pillai & Subbarao, 2008); (Arkat, et al., 2011); (Khaksar-Haghani, et al., 2013), Tabu Search (Logendran & Ramakrishna, 1995); (Adenso-Diaz, et al., 2001), Simulated Annealing (Chen, et al., 1995); (Zolfaghari & Liang, 2002); (Xambre & Vilarinho, 2003), Ant Colony Optimization (ACO) (Spiliopoulos & Sofianopoulou, 2008); (Solimanpur, et al., 2010), Particle Swarm Optimization (PSO) (Durán, et al., 2008); (Anvari, et al., 2010), Bee's Algorithm (Pham, et al., 2006), Water Flow-Like Algorithm (Wu, et al., 2010), Firefly-Inspired Algorithm (Sayadi, et al., 2013), Bacteria Foraging Algorithms (Nouri & Hong, 2013), Bat Algorithms (Soto, et al., 2016); (Olivares, et al., 2018) etc. Few noticeable facts in CM literature,

- Researchers preferred to consider the binary data instead of workload data in CM research.
- Ratio data based researches considered optimization of the cell load variations and number of bottleneck machines and weighted sum method is used to solve the problem which eventually reduces the problem into a scalarized single-objective problem. Due to that reason, the opted methodologies are not multi-objective in true sense.

- Not many performance metrics are available in past literature except the recently published one ((Ghosh, et al., 2017).

In this paper an attempt is made to develop a bi-objective model of CFP considering the percentage utilization of machines. Due to the novelty in the problem model, a new data generation algorithm is proposed which efficiently generates test data. A Non-dominated Sorting Bat Algorithm (NSBA) is implemented, which incorporates modified form of velocity and position update expressions. NSBA algorithm is compared with a published Multi-Objective Bat Algorithm (MOBA) (Yang, 2010), which exploits weighted sum approach. A detailed statistical analysis is presented to validate the proposed algorithm, which further proves the data consistency. The rest of this work proceeds in following order, problem formulation and mathematical model are presented in section #2. NSBA algorithm is presented in section #3. Section #4 describes the computational results and section #5 concludes this research.

2. Problem Formulation

The exact definition of utilization percentage based cell design problem as stated by (Ghosh, et al., 2017), is,

$$u_{ij} = \frac{(t_{ij} \times n_j)}{MH_i} \tag{1}$$

Where,

$$u_{ij} = \begin{cases} 0, & if\ part\ j\ is\ not\ processed\ by\ machine\ i \\ \sim 0, & if\ part\ j\ is\ being\ processed\ by\ machine\ i \end{cases} \tag{2}$$

$$\sum_{j=1}^{p} u_{ij} \leq 1 \tag{3}$$

t_{ij} = unit processing time (hour/unit) of part j on machine $i;\ i \in [1,q]\ and\ j \in [1,p]$
n_j = production volume of part j
MH_i = available machine hours of machine i
$U = [u_{ij}]$ is an $(p \times q)$-machine-component incidence matrix where
u_{ij} = Percentage utilization of machine i induced by part j

A machine-part utilization matrix U could be obtained from Eq. (1), which was popularly stated as the workload data. u_{ij} denotes a fraction of machining hours of i^{th} machine required to process the total volume of j^{th} part. (Ghosh, et al., 2017) classified this as “percentage utilization of machines”. The value of u_{ij} could be set as either zero or non-zero based on the processing requirement (Eq. (2)). The cumulative utilization percentage of all parts on i^{th} machine is constrained to be not greater than 1 since utilization of any machine is ideally restricted within 100%. Eq. (3) is overlooked in past literature of CMS which are decisive factors while designing the data. Therefore a new algorithm is developed to obtain data. The flowchart is furnished in Figure 1. This algorithm includes all the constraints carefully. It also influences the number of zeros in the generated matrix. The percentage of zeros would increase with the size of the matrix systematically based on the practical observation.

2.1 Bi-Objective Model

Machine utilization has never been practiced as a production factor in CM. To accomplish that goal, a new mathematical expression is deduced and presented in this study. The proposed multi-objective problem minimizes the total utilizations induced by the bottleneck machines (also known as exceptional elements), maximizes total in-cell machine utilization and minimizes total number of voids in cells. The expressions of the above three objectives are furnished hereunder.

Total utilization on exceptional elements (TEU) is expressed as,

$$minimize\ Z1 = \frac{\sum_{k=1}^{c}\sum_{j=1}^{p}\sum_{i=1}^{q}\left(x_{ik} - y_{jk}\right)^{2} u_{ij}}{2\sum_{j=1}^{p}\sum_{i=1}^{q} u_{ij}} \tag{4}$$

Total number of voids are expressed as,

$$minimize\ Z2 = \frac{\sum_{k=1}^{c}\sum_{j=1}^{p}\sum_{i=1}^{q}(1 - a_{ij})\, x_{ik}\, y_{jk}}{\sum_{j=1}^{p}\sum_{i=1}^{q} a_{ij}} \tag{5}$$

The weighted sum objective function *Z* of all the objectives is expressed as,

$$Minimize\ Z = w1.Z1 + w2.Z2 \tag{6}$$

Where *w1*, *w2* are the weight factors and sum of these is equals to 1. The weight factors *w1, w2,* assign different weights to the objectives. These are settled in the range [0, 1]. In experience, TEU have larger impact than number of voids. However, for simplicity, same weights are assigned to all the objectives in this work. Thus all the objectives share the same importance.

$$a_{ij} = \begin{cases} 1, if\ part\ j\ is\ processed\ in\ machine\ i, \\ 0, if\ part\ j\ is\ not\ processed\ in\ machine\ i \end{cases} \qquad i \in [1, q]\ and\ j \in [1, p] \tag{7}$$

$$x_{ik} = 1\ if\ machine\ i\ is\ in\ cell\ k, else\ 0 \qquad \forall i, k \tag{8}$$

$$y_{jk} = 1\ if\ part\ j\ is\ in\ cell\ k, else\ 0 \qquad \forall j, k \tag{9}$$

$$\sum_{k=1}^{c} x_{ik} = 1 \qquad \forall i \tag{10}$$

$$\sum_{i=1}^{q} x_{ik} \geq 1 \qquad \forall k \tag{11}$$

$$\sum_{k=1}^{c} y_{jk} = 1 \qquad \forall j \tag{12}$$

$$\sum_{j=1}^{p} y_{jk} \geq 1 \qquad \forall k \tag{13}$$

Eq. (7) depicts the machine-part incidence matrix and Eq. (8) - (9) are the decision variables. Eq. (10) - (13) are the assignment constraints, which ensure that each machine/part is assigned to only one cell and each cell holds at least one machine/part.

2.2 Performance Measure

A novel performance measure is recently proposed which is known as Utilization-based grouping efficiency (UGE) which is a proven metric when compared with the previous metrics (Ghosh, et al., 2017). UGE can competently handle machine utilization percentage. It’s proved to be competent while optimizing TCU and TEU for the stated problem.

UGE is demonstrated as,

$$UGE = \frac{\left(\sum_{k=1}^{c}\left[U_{cell}^{k}\left(1-\frac{V_k}{E_k}\right)\right]\right)\left(1-\frac{U_{ee}}{\sum_{k=1}^{c}U_{cell}^{k}}\right)}{U_{plant}} \tag{14}$$

Where c: number of cells; k: index of cell $\{k=1,2,\dots c\}$; U_{cell}^{k}: Total utilization of k^{th} cell; U_{plant}: Total utilization of plant; U_{ee}: Total utilization outside the block diagonal cell structure; V_k: Total number of voids in cell k $\{k=1,2,\dots c\}$; E_k: Total number of elements in cell k $\{k=1,2,\dots c\}$.

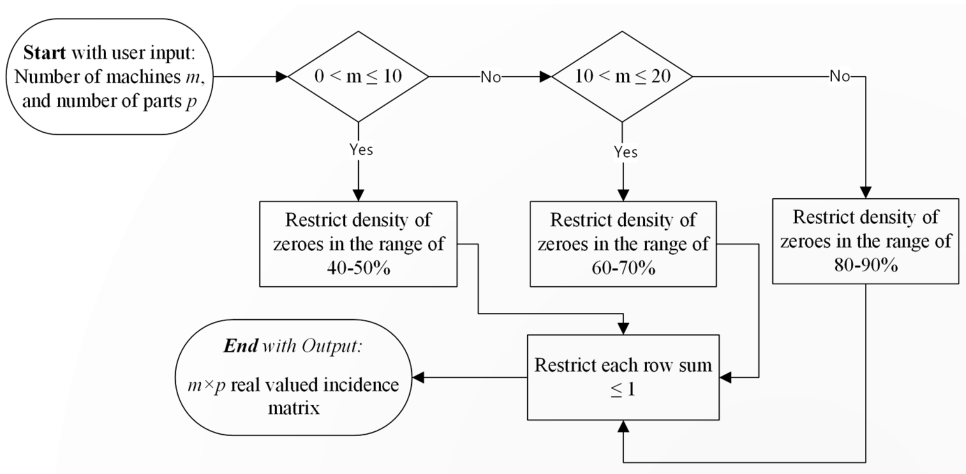

Figure 1. Data generating algorithm

3 Research Methodology

Bat inspired algorithm is (BA), a population based algorithm, proposed by (Yang, 2010). Initially it is developed to solve multi-objective optimization problems in continuous domain, which is the scalar type of multi-objective problems based on weighted sum method (Yang, 2010).
Apart from continuous optimization, the bat inspired algorithm (BA) is being practiced in discrete optimization (Marichelvam, et al., 2013), clustering of data and processing of images (Dhar, et al., 2017); (Zhang & Wang, 2012).
Among other applications, fuzzy mathematics based BA is practiced with dynamic selection of parameters while execution (Perez, et al., 2015). Chaotic BAs are exploited by some researchers in various works (Lin, et al., 2012); (Abdel-Raouf, et al., 2014); (Gandomi & Yang, 2014). Ref. (Khooban & Niknam, 2015) proposed a self-adaptive BA with fuzziness for parameter optimization of PI controller successfully. Ref. (Yılmaza & Küçüksille, 2015) has developed a modified form of BA with enhanced local and global search characteristics using three different techniques which outpace the standard BA and other published methods. Ref. (Osabaa, et al., 2017) has demonstrated a discrete BA to solve different variants of traveling salesman problems (TSP) such as symmetric and asymmetric and proposed some improvement in the basic bat algorithmic structure and tested successfully.

BA is recently being practiced in several area of production and manufacturing engineering. Ref. (Kumar, et al., 2016) utilized BA to optimize the tolerance based on parallel objectives to minimize the cost of manufacturing, present worth of expected quality loss and quality loss and obtained better results than existing methods. Ref. (Soto, et al., 2016) proposed a BA which elucidate production cell design problem with near-optimum values for 94% of the test data. The algorithm has shown excellent results and high convergence rates in their research. Ref. (Tharakeshwar, et al., 2017) implemented BA based optimization on effectiveness and total cost with different schematic parameters such as baffle cuts, baffle spacings, pitch, tube length, tube layout pattern and obtained pareto solutions which can trade-off among the objectives considered. The result was successfully compared with genetic algorithm. Ref. (Olivares, et al., 2018) proposed a version of BA that can obtain its parameter values from its own experience. They have solved many instances of production cell formation problems successfully. Ref. (Dao, et al., 2018) developed a parallel variant of the BA with random-key encoding structure, special communication strategy and makespan procedure for job shop scheduling and obtained better convergence and accuracy. Only one article is found where a multi-objective model for BA, NSBAT-II is developed in the true sense (Prakash, et al., 2016). A detailed survey on various BA and its applications can be studied in the article published by (Jayabarathi, et al., 2018).

3.1 The basic BA

The algorithm is a population based algorithm which mimics the activities of microbats having the echolocation characteristics. When bats searches for foods, they show some sonic behavior by emitting sound waves (pulse) with different emission rates, wave frequencies and loudness to calculate the distance between their existing position and target position with greater efficiency. Therefore every bat can be thought of as an individual i in a population N_p having certain velocity v_i to move from current position x_i^t at time t to next position x_i^{t+1} at time $t+1$. Every bat is associated with some pulse emission rate r_i, fixed frequency f_{min} and loudness A_i. Ideally the bats are assumed to have self-adaptive nature while adjusting the pulse emission rate r_i [0, 1]. The loudness is supposed to be limited in the range [A_{min}, A_{max}]. After the parameter initialization, the main loop of the algorithm starts. In each iteration, every bat of the population flies to a new position with a velocity update using the following expressions.

$$f_i = f_{min} + (f_{max} - f_{min}) \times RN \tag{15}$$

$$v_i^t = v_i^{t-1} + [x_i^{t-1} - x_{best}] \times f_i \tag{16}$$

$$x_i^t = x_i^{t-1} + v_i^t \tag{17}$$

In Eq. (15) RN is a random number generated in the interval [0, 1]. x_{best} is the global best solution in the population. In addition to this a random walk based local search is introduced as,

$$x_i^t = x_i^{t-1} + \varepsilon \times A^t \tag{18}$$

where ε is a random number in the interval [-1,1], and A^t is the mean loudness of the population at time step t. The loudness and emission rate are updated using,

$$A_i^{t+1} = \alpha \times A_i^t \tag{19}$$

$$r_i^{t+1} = r_i^0 \times [1 - \exp(-\gamma \times t)] \tag{20}$$

where α and γ have fixed values. This value is kept ~0.9 mostly. The loudness and the pulse emission rate update takes place when an improved solution is obtained which further signifies the movement of bats towards optimal direction.

The basic BA pseudocode is,

START
Step 1. Define the fitness function $f(x)$, $X=(x_1, x_2, \ldots, x_d)^T$
Step 2. Initialize the population of microbats x_i, $(i = 1, 2, \ldots, n)$ and velocity v_i
Step 3. Define the pulse frequency f_i at x_i
Step 4. Initialize the pulse rate r_i and loudness A_i
Step 5. While (t< maximum number of iteration) do
Step 6.   for each bat x_i in the population do
Step 7.   Generate new bats using Eq. (15), (16) and (17)
Step 8.   if $rand>r_i$ then
Step 9.   Select one solution among the best ones
Step 10.   Generate a local solution using Eq. (18)
Step 11.   end
Step 12.   if $rand<ri$ and $f(xi)<f(x_{best})$ then
Step 13.   Accept the new solution
Step 14.   Increase r_i and reduce A_i
Step 15.   end
Step 16.   end
Step 17. end
Step 18. return the global best bat x_{best}
STOP

This algorithm is designed to optimize the scalar form of multi-objective functions. That is more precisely the combined form of objectives using weighted sum approach. However, in such cases, the obtained solution quality will be substantially compromised due to the natural shortcomings of weighted sum method. To address this issue, another variant of BA is used in this study which exploits non-dominated searching for Pareto optimality. The simple BA is modified enough to incorporate discretization and improved local search method for faster convergence. This new method is known as NSBA, which follows the similar procedure of NSGA 2 by (Deb, et al., 2002).

3.2 The NSBA algorithm

The proposed NSBA starts with a population of N_p bats. NSBA is real coded and depends on the pre-defined number of cells. Each solution has a position x_i where i is the index in the population matrix. Every row in the matrix represents a bat. If the number of machines and parts are m and n respectively, then the bat is represented by $m+n$ bits vector. Every bat is associated with its velocity v_i which is also a $m+n$ bits vector.

3.2.1 Initial population generation and encoding scheme

Initial population is generated using a specifically designed random vector generator function which states that every solution vector has a length of $m+n$ and the elements of the vector are in the interval [1, c] with at least one occurrence (Boulif & Atif, 2006). An example solution vector of a 6 machines, 8 parts and 2 cells test problem has the length of 14 bits (6+8) with each bit representing a cell number (either 1 or 2). Encoding of a solution is shown in Table 1.

Table 1. Example bat of NSBA for the 6×8 problem

M1	M2	M3	M4	M5	M6	P1	P2	P3	P4	P5	P6	P7	P8
1	2	2	1	2	2	1	2	1	2	1	2	1	2

This implies cell 1 contains machines 1, 4 and parts 1, 3, 5, 7 and cell 2 contains machine 2, 3, 5, 6 and parts 2, 4, 6, 8 respectively. Using this method the whole initial population matrix of size $N_p\times(m+n)$ is generated where N_p is the number of bats in the population.

3.2.2 Initialization of parameters

The choice of parameters in population based algorithms can have a large impact on the process of optimization. Selection of optimum parameters is a critical task for the researchers, which can attain global optimal solution (Taherkhani & Safabakhsh, 2016). In this study, after rigorous testing, the parameters of NSBA are prefixed with the following values, number of iterations = 1000, population size = 500, constant loudness $A^0 = 0.25$, constant rate of emission $r^0 = 0.5$, minimum frequency $f_{min} = 0$, maximum frequency $f_{max} = 2$, $\alpha = \gamma = 0.9$

3.2.3 Objective functions

The objective function principally evaluates the fitness of a solution vector by computing a numerical score. Since the utilization based CFP has two objectives $Z1$, and $Z2$ (Eq. (4)-(5)), therefore these functions are used to check fitness of the solutions achieved by stated multi-objective algorithm. Each solution is then checked with its next solution in the population for the non-dominance. The solution is marked to be strictly non-dominated if it is superior for all the objectives considered. Thereafter the non-dominated solution is moved to an empty pool matrix. This procedure is repeated for every solution in the population. At the end, a pool of non-dominated solutions is generated. Unlike single objective optimization, Multi-objective problems cannot possess a single global best solution. All the solutions marked as non-dominated, represent the optimal or global best solutions.

3.2.4 Velocity and position update strategy for the bats

In this step, velocity and position update strategies are defined. x_i^{t-1} is updated using the Eq. (16)-(17), which are modified in Eq. (21) and Eq. (22). x_i^{t-1} and x_{best} can be expressed as assignment matrices y_i^{t-1} and y_{best} of size $(m+n)\times c$ where c is the number of cells and m, n are the number of machines and parts respectively,

$$v_i^t = v_i^{t-1} + [y_i^{t-1} - y_{best}] \times f_i \tag{21}$$

$$y_i^t = y_i^{t-1} \times v_i^t \tag{22}$$

Eq. (22) generates an intermediate assignment matrix $y_i^t = [a_{k\times j}]_{(m+n)\times c}$ $\{1\leq k\leq m+n; 1\leq j\leq c\}$ with real values. In order to obtain the equivalent binary assignment matrix $y\_bin_i^t = [a\_bin_{k\times j}]_{(m+n)\times c}$ $\{1\leq k\leq m+n; 1\leq j\leq c\}$, some assignment rules are applied as,

$$a\_bin_{k\times j} = \begin{cases} 1, & a_{k\times j} == \max(a_{k\times j})) \quad and\ a_{i\times j} \neq 0 \quad 1 \le j \le c \\ 1, & (a_{k\times j} < 0) \text{ and } \max(a_{k\times j}) == 0 \quad 1 \le j \le c \\ 1, & (a_{k\times j} == 0\ and \min no.of\ non - zero\ in\ j^{th} cell \quad 1 \le j \le c \\ 0, & Orherwise \end{cases} \tag{23}$$

This strategy could be illustrated using the 6×8 example CFP.

3.2.4.1 Illustration

The x_i^{t-1} is,

M1	M2	M3	M4	M5	M6	P1	P2	P3	P4	P5	P6	P7	P8
1	2	2	1	2	2	1	2	1	2	1	2	1	2

The x_{best} is,

M1	M2	M3	M4	M5	M6	P1	P2	P3	P4	P5	P6	P7	P8
1	1	1	1	2	2	1	1	1	2	1	1	1	2

y_i^{t-1} is,

	C1	C2
M1	1	0
M2	0	1
M3	0	1
M4	1	0
M5	0	1
M6	0	1
P1	1	0
P2	0	1
P3	1	0
P4	0	1
P5	1	0
P6	0	1
P7	1	0
P8	0	1

y_{best} is,

	C1	C2
M1	1	0
M2	1	0
M3	1	0
M4	1	0
M5	0	1
M6	0	1
P1	1	0
P2	1	0
P3	1	0
P4	0	1
P5	1	0
P6	1	0
P7	1	0
P8	0	1

The velocity vector v_i^{t-1} is

M1	M2	M3	M4	M5	M6	P1	P2	P3	P4	P5	P6	P7	P8
-0.57	-0.199	-0.37	0	0.87	0	0.281	0.243	0	-0.623	0	-0.044	-0.26	-0.42

RN=0.1270; fmin = 0; fmax = 2;

Hence the value of f_i is 0.2540 from Eq. (15), the value of v_i^t from Eq. (21) is,

M1	M2	M3	M4	M5	M6	P1	P2	P3	P4	P5	P6	P7	P8
-0.072	0.229	0.21	0	0.11	0	0.036	0.28	0	-0.079	0	0.25	-0.033	-0.053

The intermediate assignment matrix of new position is y_i^t computed from Eq. (22), which is,

	C1	C2
M1	-0.072382485287	0
M2	0	0.2287032561446
M3	0	0.2069885105584
M4	0	0
M5	0	0.1104785301753
M6	0	0
P1	0.0356832953785	0
P2	0	0.2848314289463
P3	0	0
P4	0	-0.079112786551
P5	0	0
P6	0	0.2483862126701
P7	-0.033016572236	0
P8	0	-0.053334462843

The binary assignment matrix $y\_bin_i^t$ computed from rules stated in Eq. (23)

	C1	C2
M1	1	0
M2	0	1
M3	0	1
M4	1	0
M5	0	1
M6	1	0
P1	1	0
P2	0	1
P3	1	0
P4	0	1
P5	1	0
P6	0	1
P7	1	0
P8	0	1

Finally it is possible to interpret the new solution vector x_i^{t-1} from the binary matrix $y\_bin_i^t$, which is,

M1	M2	M3	M4	M5	M6	P1	P2	P3	P4	P5	P6	P7	P8
1	2	2	1	2	1	1	2	1	2	1	2	1	2

3.2.5 Swap based local search

In order to improve the speed of convergence, a small local search is performed which might explore the unexplored area of solution search space. This part of the algorithm is an addition to the method which has nothing to do with the behavior of the bats. A probability P_{local} is considered with the generation of a random number *RN1*. If $RN1<P_{local}$, a non-dominated solution vector is selected from the pool and perform two-point random swap operations on selected elements of the vector. This procedure would try to diversify the population with trivial modifications, which eventually helps in finding optimal solutions. The pseudocode is furnished as,

Step 1. **While** $RN1<P_{local}$
Step 2. **Do**
Step 3. **Select** vector *v(1 to m+n)* from non-dominated pool where *m, n* are numbers of machines and parts respectively
Step 4. **Define** *vm = v(1 to m)* and *vn=v(1 to n)*
Step 5. **Generate** random numbers r1 ∈ [1, m], r2 ∈ [1, m], r3 ∈ [1, n] and r4 ∈ [1, n]
Step 6. *tempmc = vm(r1)*
Step 7. *vm(r1) = vm(r2)*
Step 8. *vm(r2) = tempmc*
Step 9. *temppt = vn(r3)*
Step 10. *vn(r3) = vn(r4)*
Step 11. *vn(r4) = tempt*
Step 12. *v*=concatenate (*vm, vn*)
Step 13. **Return**

3.2.6 Termination condition

The execution of the algorithm is controlled by some stopping condition. The execution is eventually terminated if the count attains the pre-defined number of iterations.

3.2.7 Flowchart of NSBA

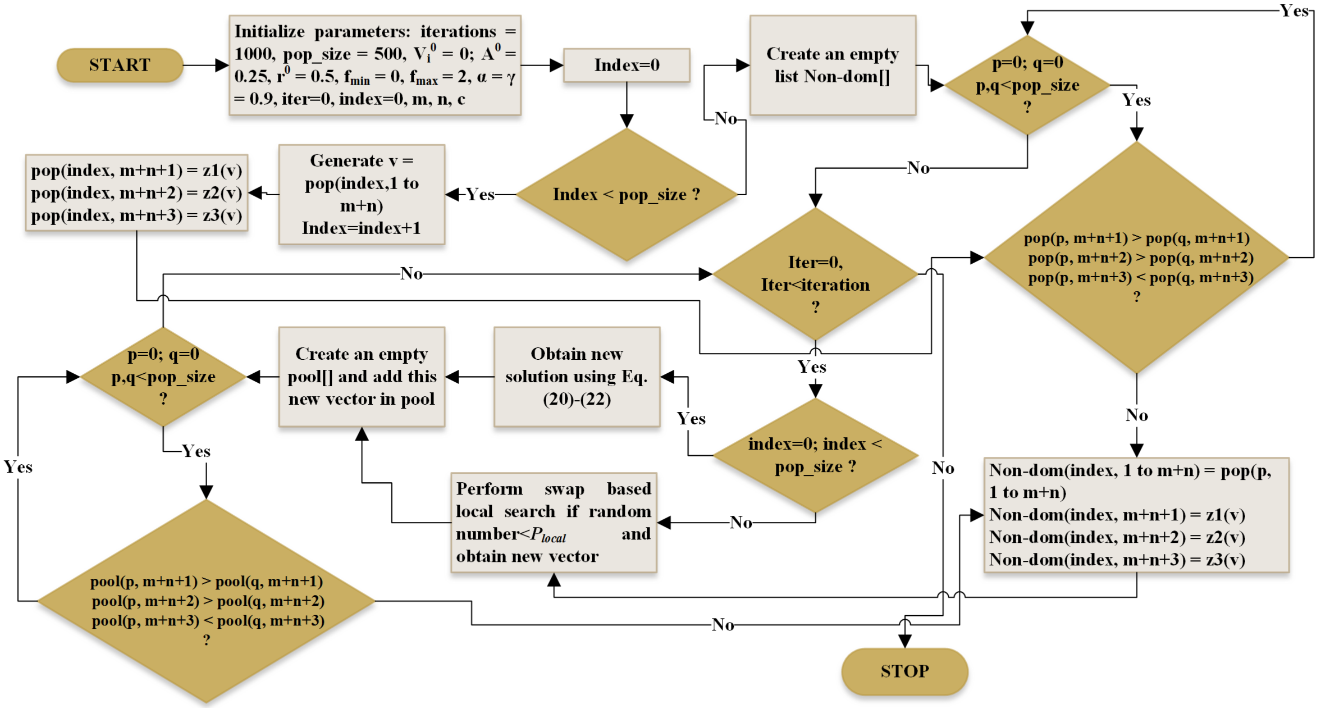

Figure 2. NSBA Flowchart

4 Results and Discussions

An intensive discussion is carried out in this section, which points out the efficiency of the proposed NSBA. The idea is to verify the convergence and solution quality obtained by NSBA and compare with MOBA. MOBA algorithm is based on the basic bat algorithm furnished in section 3.1 with modification portrayed in Eq. (21)-(23). With an aim of validating the proposed bi-objective CFP, work-load data are required. Only three datasets are available in past literature (Ghosh, et al., 2017). Available datasets are 5×7, 7×11 and 10×10 in size.

Table 2. Comparison among NSBA, MOBA and MCDM methods

Sl. No.	Size	No. of Cells	MCDT Model Based on AHP (Ghosh, et al., 2017)				NSBA				MOBA			
			No. of Voids and EE	TEU	UGE	CPU Time	No. of Voids and EE	TEU	UGE	CPU Time	No. of Voids and EE	TEU	UGE	CPU Time
1	5×7	2	Voids=6; Exceptional Elements=3	0.56	53.45	0.021	Voids=4; Exceptional Elements=2	0.1429	55.56	35.6488	Voids=4; Exceptional Elements=2	0.1429	55.56	46.4578
2	7×11	3	Voids=16; Exceptional Elements=7	0.91	37.13	0.066	Voids=10; Exceptional Elements=6	0.1562	43.29	39.6673	Voids=10; Exceptional Elements=6	0.2172	43.29	49.1042
3	10×10	3	Voids=15; Exceptional Elements=1	0.31	60.76	0.104	Voids=10; Exceptional Elements=0	0	70.59	40.688	Voids=10; Exceptional Elements=0	0	70.59	43.5961

Therefore more utilization based datasets are generated using the data generating algorithm described earlier (Figure 1). 11 datasets of small to medium sizes ranging from 4×7 to 10×10 are obtained. The proposed NSBA algorithm and the other variant of bat algorithm are implemented with MATLAB libraries on Intel 8650U @1.90 GHz laptop. The results are compared with the published results and the NSBA is shown to outperform the published Multi-Criteria Decision Method (MCDM) technique and MOBA. The results for those three datasets are displayed in Table 2. Due to the nature of NP-Hardness, obtaining solutions is not an easy task. The number of variables and constraints increases with the size of the data.

UGE is used as a performance metric in this study (Ghosh, et al., 2017). The evaluation criteria of NSBA and MOBA are based on TEU, and the total number of voids for the bi-objective CFP. For NSBA algorithm, the presented solutions are picked from Pareto front. The results obtained for generated data are depicted in Table 3. Both the variants of BA are shown to obtain good results. Furthermore these results reveal few significant aspects related to the objective functions and UGE. For test data 1 and 4 both MOBA and NSBA yield global best solutions. For test data 3, 6, 8, 9, MOBA obtains near best solutions, which are trivially better than the NSBA solutions. Whereas NSBA attains better solutions for the rest of the 6 test data. For all these 6 test data NSBA not only shows better UGE values but TEU value is also minimized. Therefore it can be stated that the NSBA is 35.71% better than MOBA algorithm for all the 14 test data. However, influence of voids count is trivial in this study, instead TEU has greater impact due to the design strategy of UGE. Thus the solutions attained by NSBA sometime depict more number of voids than the MOBA solutions. It is also observed that some of the results obtained by MOBA, hold better UGE scores but they are inferior in terms of TEU due to the assigned weight factors to the objectives. Since both the objectives are equally treated, thus the number of voids receives same importance in this study. Therefore the objective values are reduced with an improved UGE. This fact indicates the practical limitations and inferiorities of weighted sum method. To prove the competence of NSBA over MOBA, some statistical analyses are performed based on the obtained UGE scores for all the 14 test data.

Table 3. Experimental results of NSBA and MOBA on 11 test data obtained using data generation algorithm

			NSBA					MOBA				
No.	**Size**	**No. of Cells**	**UGE**	**TCU**	**TEU**	**Voids**	**CPU Time**	**UGE**	**TCU**	**TEU**	**Voids**	**CPU Time**
1	4×7	2	**53.93**	0.8187	0.1813	2	33.165	**53.93**	0.8187	0.1813	2	31.9222
2	5×10	2	**47.15**	0.86438	0.13562	12	37.70	43.94	0.7775	0.2225	5	39.6382
3	6×8	2	38.57	0.80463	0.2281	7	37.73	**39.17**	0.7719	0.19537	10	37.7468
4	7×10	2	**46.74**	0.78377	0.21623	6	37.59	**46.74**	0.7838	0.2162	6	36.8194
5	7×11	2	**49.18**	0.8411	0.1589	12	38.62	48.47	0.8343	0.1657	11	39.1243
6	8×15	2	42.12	0.8767	0.1233	41	39.80	**45.2**	0.8598	0.1402	29	42.7491
7	8×22	2	**46.05**	0.8914	0.1086	56	44.51	35.33	0.8246	0.1754	62	47.850
8	9×9	2	46.15	0.8822	0.1178	21	40.58	**47.64**	0.8286	0.1714	11	37.3694
9	9×15	2	47.74	0.845	0.154	21	40.56	**48.71**	0.915	0.085	38	35.8894
10	10×10	3	**40.07**	0.8149	0.1851	16	64.93	27.7	0.72428	0.27572	13	36.840
11	10×10	3	**33.16**	0.7487	0.2513	11	68.45	26.07	0.737	0.263	18	93.6095

First, the normality of data is tested using Anderson-Darling normality test. The plots are shown in Figure 3. The null hypothesis used is H0, if the data is normal. It implies the rejection of the null hypothesis when p-value $\leq \alpha$, which is the significance level of 0.05. We accept the null hypothesis since, the dots fit the trend line on the normal probability plots, for NSBA, p-value=0.127>0.05 and for MOBA P-value=0.606>0.05. Therefore, the null hypothesis is accepted. Hence it can be concluded with 95% confidence level that the data follow normal distribution.

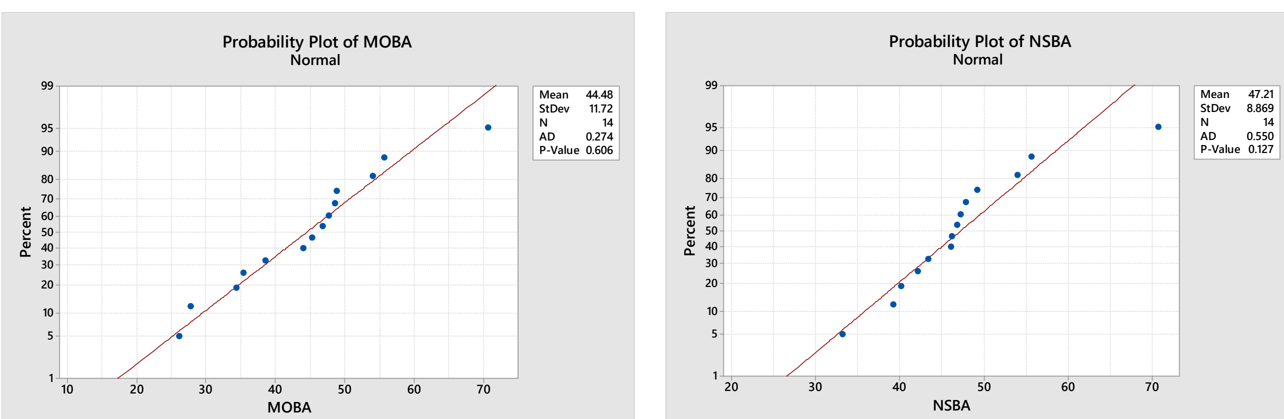

Figure 3. Normality plots of both MOBA and NSBA

These data are considered as two independent sets of values. Therefore the equality of variances is tested with 2 variances f-test with σ_1: standard deviation of MOBA; σ_2: standard deviation of NSBA. The ratio σ_1/σ_2 is the indicator. If the test statistic < critical value ($F < F_{critical}$) accept the null hypothesis; in other words, if p-value>α, accept the null hypothesis.

Table 4. f-test results for equality of variances for MOBA and NSBA

	MOBA	*NSBA*
Mean	44.48142857	47.2071429
Variance	137.390567	78.6660681
Observations	14	14
df	13	13
F	1.746503547	
P(F<=f) one-tail	0.163549506	
F Critical one-tail	2.576927084	

The test results are shown in Table 4. According to the results, it is clearly visible that the $F < F_{critical}$ (1.746503547<2.576927084); p-values > α (0.163549506>0.05. Therefore the null hypothesis is accepted and it is concluded that the variances are equal. Further the t-test is performed assuming equal variances. The result of t-test is reported in Table 5.

Table 5. t-test results assuming equal variances for MOBA and NSBA

	MOBA	*NSBA*
Mean	44.48142857	47.2071429
Variance	137.390567	78.6660681
Observations	14	14
Pooled Variance	108.0283176	
Hypothesized Mean Difference	0	
df	26	
t Stat	-0.693841934	
P(T<=t) one-tail	0.246967168	
t Critical one-tail	1.70561792	
P(T<=t) two-tail	0.493934336	
t Critical two-tail	2.055529439	

If the test statistic < critical value ($F < F_{critical}$) accept the null hypothesis; in other words, if p-value>α, accept the null hypothesis. Since the null hypothesis is that the mean difference=0, therefore this would be decided with a two-sided test. Two-tail values are used for the analysis. According to Table 5, the test statistic< critical values (-0.693841934 < 1.70561792 and -0.693841934 < 2.055529439) and the p-values for one-tail and two-tail > α (0.246967168 > 0.05 and 0.493934336 > 0.05), thus the null hypothesis is accepted and the means are same. Therefore, the obtained results are consistent. Thus the MOBA and NSBA are equally good and not differentiable to a large extent. More specifically, it could be stated that, even though the NSBA (mean= 47.2071429) outpaces MOBA (mean= 44.48142857), they are equally capable of producing improved results.

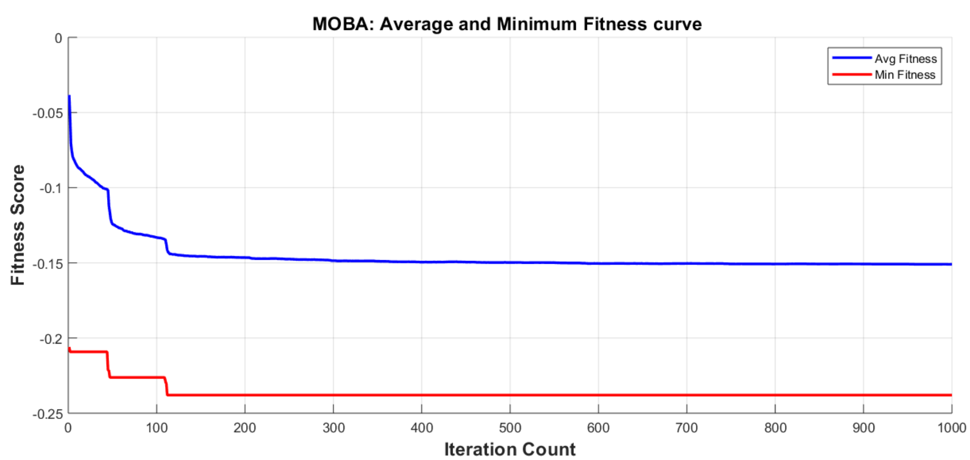

Figure 4. Convergence curve of MOBA algorithm for 6×8 test data

4.1 Convergence analysis and Pareto front

Convergence properties for MOBA are nearly similar for all the test data. For an example, the problem #3 (Table 3) of size 6×8 is selected to depict the convergence property of the MOBA algorithm (Figure 4). Figure 5 portrays the Pareto plot for NSBA over two objectives for 6×8 data. The Pareto front contains 15 solution points, which are equally good. The most promising one is picked and suggested in Table 3. It also depicts the conflicting behavior of the objectives.

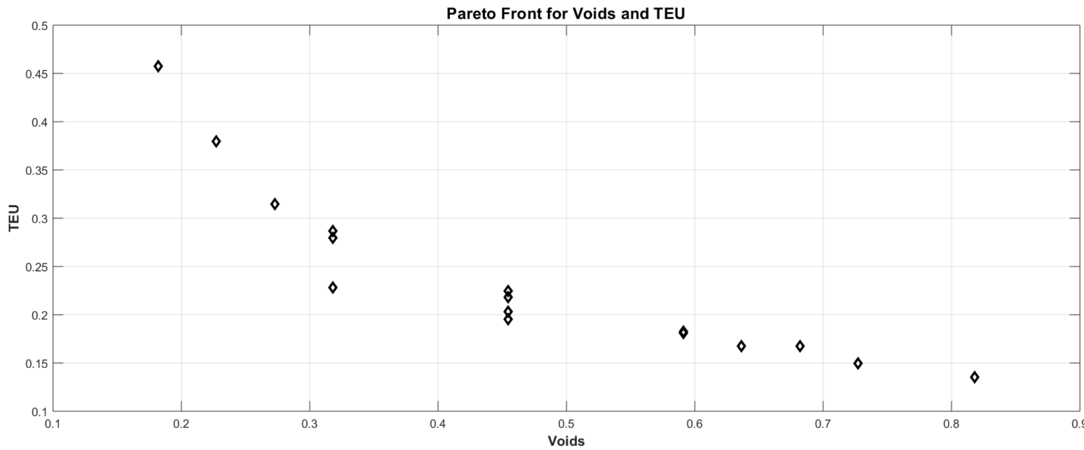

Figure 5. Pareto front of NSBA for 6×8 data

5 Conclusions

This article inspects some crucial aspects of cell formation problem based on percentage utilization of machines. A bi-objective mathematical expression is derived in this study which carefully minimizes number of voids and cumulative utilization induced by bottleneck machines. Further, a novel multi-objective algorithm namely NSBA is successfully implemented for CFP. The performance of NSBA is compared with published MOBA and tested on 14 problems. NSBA algorithm obtains good solutions quickly and outpaces MOBA. The obtained results are validated using Anderson-Darling's normality test and two more statistical tests (f-test and t-test). Following conclusions are derived,

- Proposed data generating algorithm generates real-world-like data, which could be used for Percentage utilization based CFPs.
- Proposed bi-objective mathematical model could be effectively used as fitness functions for metaheuristic algorithms.
- Pareto solutions obtained using multi-objective techniques for CFP are more effective than solutions obtained using weighted sum approach for multi-objective problems.
- NSBA is an effective technique, which outpace MOBA by 35.7% for 14 utilization based CFPs considered.
- Statistical tests confirm the competence of NSBA, which further prove the feasibility of the stated mathematical model.

This work would be extended in future with another production factor based on part routing information and testing would be done on real-world industrial data.

References

Abdel-Raouf, O., Abdel-Baset, M. & El-Henawy, I., 2014. An improved chaotic bat algorithm for solving integer-programming problems. *International Journal of Modern Education and Computer Science,* 6(8), pp. 18-24.

Adarsh, B. R., Raghunathan, T., Jayabarathi, T. & Yang, X. S., 2016. Economic dispatch using chaotic bat algorithm. *Energy,* Volume 96, pp. 666-675.

Adenso-Diaz, B., Lozano, S., Racero, J. & Guerrero, F., 2001. Machine cell formation in generalized group technology. *Computers & Industrial Engineering,* 41(2), p. 227–240.

Anvari, M., Mehrabad, M. S. & Barzinpour, F., 2010. Machine–part cell formation using a hybrid particle swarm optimization. *International Journal of Advanced Manufacturing Technology,* 47(5), pp. 745-754.

Arkat, J., Hosseini, L. & Farahani, M. H., 2011. Minimization of exceptional elements and voids in the cell formation problem using a multi-objective genetic algorithm. *Expert Systems with Applications,* 38(8), p. 9597–9602.

Arora, P. K., Haleem, A. & Singh, M. K., 2013. Recent development of cellular manufacturing systems. *Sadhana,* 38(3), pp. 421-428.

Boulif, M. & Atif, K., 2006. A new branch-&-bound-enhanced genetic algorithm for the manufacturing cell formation problem. *Computers & Operation Research,* 33(8), pp. 2219-2245.

Chandrasekaran, M. P. & Rajagopalan, R., 1986. MODROC: an extension of rank order clustering for group technology. *International Journal of Production Research,* 24(5), pp. 1221-1233.

Chattopadhyay, M., Dan, P. K. & Mazumdar, S., 2014. Comparison of visualization of optimal clustering using self-organizing map and growing hierarchical self-organizing map in cellular manufacturing system. *Applied Soft Computing,* Volume 22, pp. 528-543.

Chattopadhyay, M. et al., 2013. Neuro-genetic impact on cell formation methods of cellular manufacturing system design: a quantitative review and analysis. *Computers & Industrial Engineering,* 64(1), pp. 256-272.

Chen, C. L., Cotruvo, N. A. & Baek, W., 1995. A simulated annealing solution to the cell formation problem. *International Journal of Production Research,* 33(9), p. 2601–2614.

Dao, T. K., Pan, T. S., Nguyen, T. T. & Pan, J. S., 2018. Parallel bat algorithm for optimizing makespan in job shop scheduling problems. *Journal of Intelligent Manufacturing,* 29(2), p. 451–462.

Deb, K., Pratap, A., Agarwal, S. & Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation,* 6(2), pp. 182-197.

Dhar, S. et al., 2017. A novel method for edge detection in a gray image based on human psychovisual phenomenon and Bat algorithm. In: D. Guha, B. Chakraborty & H. S. Dutta, eds. *Computer, Communication and Electrical Technology.* s.l.:Taylor & Francis Group, pp. 3-8.

Dimopoulos, C. & Mort, N., 2001. A hierarchical clustering methodology based on genetic programming for the solution of simple cell-formation problems. *International Journal of Production Research,* 39(1), pp. 1-19.

Dimopoulos, C. & Zalzala, A. M. S., 2000. Recent developments in evolutionary computation for manufacturing optimization: problems, solutions, and comparisons.. *IEEE Transactions on Evolutionary Computation,* 4(2), pp. 93-113.

Durán, O., Rodriguez, N. & Consalter, L. A., 2008. *A PSO-based clustering algorithm for manufacturing cell design.* s.l., s.n., pp. 72-75.

Gandomi, A. H. & Yang, X. S., 2014. Chaotic bat algorithm. *Journal of Computational Science,* Volume 5, pp. 224-232.

George, A. P., Rajendran, C. & Ghosh, S., 2003. An analytical-iterative clustering algorithm for cell formation in cellular manufacturing systems with ordinal-level and ratio-level data. *International Journal of Advanced Manufacturing Technology,* 22(1), pp. 125-133.

Ghosh, T., Doloi, B. & Dan, P. K., 2017. Utilization-based grouping efficiency and multi-criteria decision approach in designing of manufacturing cells. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture,* 231(3), pp. 505-522.

Ghosh, T., Sengupta, S., Chattopadhyay, M. & Dan, P. K., 2011. Meta-heuristics in cellular manufacturing: A state-of-the-art review. *International Journal of Industrial Engineering Computation,* 2(1), p. 87–122.

Goldengorin, B., Krushinsky, D. & Pardalos, P. M., 2013. *Cell Formation in Industrial Engineering: Theory, Algorithms and Experiments..* s.l.:Springer Optimization and Its Applications, Springer..

Gupta, Y., Gupta, M., Kumar, A. & Sundaram, C., 1996. A genetic algorithm-based approach to cell composition and layout design problems. *International Journal of Production Research,* 34(2), p. 447–482.

Jayabarathi, T., Raghunathan & H., G. A., 2018. The Bat Algorithm, Variants and Some Practical Engineering Applications: A Review. In: Y. X. S., ed. *Nature-Inspired Algorithms and Applied Optimization. Studies in Computational Intelligence.* s.l.:Springer, pp. 313-330.

Khaksar-Haghani, F., Kia, R., Mahdavi, I. & Kazemi, M., 2013. A genetic algorithm for solving a multi-floor layout design model of a cellular manufacturing system with alternative process routings and flexible configuration. *International Journal of Advanced Manufacturing Technology,* 66(5), p. 845–865.

Khooban, M. H. & Niknam, T., 2015. A new intelligent online fuzzy tuning approach for multi area load frequency control: Self Adaptive Modified Bat Algorithm. *International Journal of Electrical Power & Energy Systems,* Volume 71, pp. 254-261.

King, J. R., 1980. Machine-component grouping in production flow analysis: an approach using a rank order-clustering algorithm. *International Journal of Production Research,* 18(2), pp. 213-232.

Kumar, R. L., Padmanaban, K. P., Kumar, G. S. & Balamurugan, C., 2016. Design and optimization of concurrent tolerance in mechanical assemblies using bat algorithm. *Journal of Mechanical Science and Technology,* 30(6), pp. 2601-2614.

Lin, J. H., Chou, C. W., Yang, C. H. & Tsai, H. L., 2012. A chaotic levy flight bat algorithm for parameter estimation in nonlinear dynamic biological systems. *Computer and Information Technology,* Volume 2, pp. 56-63.

Logendran, R. & Ramakrishna, P., 1995. Manufacturing cell formation in the presence of lot splitting and multiple units of same machine. *International Journal of Production Research,* 33(3), p. 675–693.

Mahapatra, S. S. & Pandian, R. S., 2008. Genetic cell formation using ratio level data in cellular manufacturing systems. *International Journal of Advanced Manufacturing Technology,* 38(5), pp. 630-640.

Marichelvam, M., Prabaharan, T., Yang, X. S. & Geetha, M., 2013. Solving hybrid flow shop scheduling problems using bat algorithm. *International Journal of Logistics Economics and Globalisation,* 5(1), pp. 15-29.

Mccormick, W. T., Schweitzer, J. P. J. & White, T. W., 1972. Problem decomposition and data reorganization by a clustering technique. *Operations Research,* 20(5), pp. 993-1009.

Nouri, H. & Hong, T. S., 2013. Development of bacteria foraging optimization algorithm for cell formation in cellular manufacturing system considering cell load variations. *Journal of Manufacturing Systems,* 32(1), p. 20–31.

Olivares, R., Soto, R. & Crawford, B., 2018. *Performance evaluation of the parameterless bat algorithm to solve the manufacturing cell design problem..* s.l., IEEE.

Osabaa, E. et al., 2017. An improved discrete bat algorithm for symmetric and asymmetric Traveling Salesman Problems. *Engineering Applications of Artificial Intelligence,* Volume 48, pp. 59-71.

Papaioannou, G. & Wilson, J. M., 2010. The evolution of cell formation problem methodologies based on recent studies (1997–2008): Review and directions for future research.. *European Journal of Operational Research,* 209(3), pp. 509-521.

Perez, J., Valdez, F. & Castillo, O., 2015. A new bat algorithm with fuzzy logic for dynamical parameter adaptation and its applicability to fuzzy control design. In: O. Castillo & P. Melin, eds. *Fuzzy Logic Augmentation of Nature-Inspired Optimization Metaheuristics.* s.l.:Springer, p. 65–79.

Pham, D. T. et al., 2006. *The bees algorithm – A novel tool for complex optimization problems.* s.l., s.n., p. 454–461.

Pillai, M. V. & Subbarao, K., 2008. A robust cellular manufacturing system design for dynamic part population using a genetic algorithm. *International Journal of Production Research,* 46(18), p. 5191–5210.

Prakash, S., Trivedi, V. & Ramteke, M., 2016. An elitist non-dominated sorting bat algorithm NSBAT-II for multi-objective optimization of phthalic anhydride reactor. *International Journal of Systems Assurance Engineering,* 7(3), p. 299–315.

Sayadi, M. K., Hafezalkotob, A. & Naini, S. G. J., 2013. Firefly-inspired algorithm for discrete optimization problems: an application to manufacturing cell formation. *Journal of Manufacturing Systems,* 32(1), p. 78–84.

Selim, M. S., Askin, R. G. & Vakharia, A. J., 1998. Cell formation in group technology: review evaluation and directions for future research. *Computers and Industrial Engineering,* 34(1), p. 3–20.

Sengupta, S., Ghosh, T. & Dan, P. K., 2011. FAKMCT: Fuzzy ART K-Means Clustering Technique: a hybrid neural network approach to cellular manufacturing systems. *International Journal of Computer Integrated Manufacturing,* 24(10), pp. 927-938.

Solimanpur, M., Saeedi, S. & Mahdavi, I., 2010. Solving cell formation problem in cellular manufacturing using ant-colony-based optimization. *International Journal of Advanced Manufacturing Technology,* 50(9), pp. 1135-1144.

Soto, R. et al., 2016. *Solving Manufacturing Cell Design Problems by Using a Bat Algorithm Approach.* s.l., s.n., pp. 184-191.

Spiliopoulos, K. & Sofianopoulou, S., 2008. An efficient ant colony optimisation system for the manufacturing cells formation system. *International Journal of Advanced Manufacturing Technology,* 36(5), p. 589–597.

Srinivasan, G., 1994. A clustering algorithm for machine cell formation in group technology using minimum spanning trees.. *International Journal of Production Research,* 32(9), pp. 2149-2158.

Taherkhani, M. & Safabakhsh, R., 2016. A novel stability-based adaptive inertia weight for particle swarm optimization. Volume 38, pp. 281-295.

Tharakeshwar, T. K., Seetharamub, K. N. & Durga Prasad, B., 2017. Multi-objective optimization using bat algorithm for shell and tube heat exchangers. *Applied Thermal Engineering,* Volume 110, pp. 1029-1038.

Venugopal, V. & Narendran, T. T., 1992. Cell formation in manufacturing systems through simulated annealing: An experimental evaluation. *European Journal of Operational Research,* 63(3), pp. 409-422.

Wu, T. H., Chung, S. H. & Chang, C. C., 2010. A water flow-like algorithm for manufacturing cell formation problems. *European Journal of Operational Research,* 205(2), p. 346–360.

Xambre, A. R. & Vilarinho, P. M., 2003. A simulated annealing approach for manufacturing cell formation with multiple identical machines. *European Journal of Operational Research,,* 151(2), p. 434–46.

Yang, X. S., 2010. A New Metaheuristic Bat-Inspired Algorithm. In: G. J. R., et al. eds. *Nature Inspired Cooperative Strategies for Optimization: Studies in Computational Intelligence.* Berlin, Heidelberg: Springer, pp. 65-74.

Yılmaza, S. & Küçüksille, E. U., 2015. A new modification approach on bat algorithm for solving optimization problems. *Applied Soft Computing,* Volume 28, pp. 259-275.

Zhang, J. W. & Wang, G. G., 2012. Image matching using a bat algorithm with mutation. *Applied Mechanics and Materials,* Volume 203, p. 88–93.

Žilinskas, J., Goldengorin, B. & Pardalos, P. M., 2014. Pareto-optimal front of cell formation problem in group technology. *Journal of Global Optimization,* 61(1), pp. 91-108.

Zolfaghari, S. & Liang, M., 2002. Comparative study of simulated annealing, genetic algorithms and tabu search for solving binary and comprehensive machine-grouping problems. *International Journal of Production Research,* 40(9), p. 2141–2158.

Zolfaghari, S. & Liang, M., 2003. A new genetic algorithm for the machine/part grouping problem involving processing times and lot sizes. *Computers & Industrial Engineering,* 45(4), pp. 713-731.