Journal of Scientific & Industrial Research
Vol. 83, July 2024, pp. 748-760
DOI: 10.56042/jsir.v83i7.3871

A Hybrid PSO Based Algorithm for Solving the Machine-Part Cell Formation Problem

Adnan Tariq[1]*, Waqas Javaid[1], Waseem Shahzad[2], Muhammad Yasir[1] & Shahid Iqbal[1]

[1]Mechanical Engineering Department, [2]Mechatronics Engineering Department, Wah Engineering College, University of Wah, 4040, Pakistan

Received 11 July 2023; revised 22 April 2024; accepted 13 June 2024

Cellular Manufacturing (CM), an approach primarily based on the concept of Group Technology (GT), is one of the recent trends that help the manufacturing industry in reducing manufacturing cost and increasing productivity while maintaining quality. The idea of manufacturing parts in dedicated cells is beneficial as it results in increased manufacturing quality and reduced lead times. However, implementation of such a system in a real-life situation is always a challenging task. To overcome this challenge, several techniques, including AI-based approaches, have been developed over the years and regularly reported in literature. A very small portion of these approaches are utilizing Particle Swarm Optimization (PSO) in standard or hybrid form, whereas a larger chunk is either GA-based or utilizing other heuristics. To test the effectiveness of PSO while handling the Machine-Part Cell Formation (MPCF) problem in a CM environment, initially a standard PSO is developed during this research. Later, the same is hybridized with a Local Search Heuristic (LSH). The results of both standard and hybrid PSOs, developed during this research, are compared with the corresponding GA based methodologies, already available in literature. Computational results show that the GA based approaches have been outperformed both in terms of accuracy and computational effort. Further comparison of the results generated by the Hybrid PSO (HPSO) with several other techniques also shows that HPSO is either more or, in few cases, equally effective.

Keywords: Cellular manufacturing, Genetic algorithms, Local search heuristic, Meta-heuristics, Particle swarm optimization

Introduction

The current manufacturing industries have been consistently facing the challenges of diverse customer demands and ever-decreasing lead times while maintaining quality at a reduced cost of manufacturing. Cellular Manufacturing System (CMS), a concept based on Group Technology (GT), is a promising alternative as it offers the flexibility of a job-shop and at the same time a faster rate of production like a flow line. A successful implementation of CMS results in several advantages, however, its main contribution is product variety with shorter lead times. Despite being beneficial in many ways, its implementation is a tough challenge for the practitioners to handle. The major task in converting an existing system to CMS is distribution of parts into different part families and then grouping of machines to process each part family in a separate cell. Ideally, a part family is supposed to utilize the resources (machines) allocated to it in its respective cell, however, in practice parts in family may require processing in more than one cell resulting in intercellular moves that cause an increase in material handing cost. A larger number of intercellular moves in a CMS translate into a larger amount of overall material handling cost and consequently, lesser percentage of Grouping Efficacy (GE). This is the reason that researchers have been consistently trying to develop techniques that can solve the 35 benchmark Machine-Part Cell Formation (MPCF) problems, already available in literature, with reduced number of intercellular moves and higher percentage of GE.

*Author for Correspondence
E-mail: adnan.tariq@wecuw.edu.pk

Literature Review

Cellular Manufacturing (CM) has been largely considered as the way forward to help the manufacturing firms in facing the current challenges of increased product variety and reduced manufacturing lead times.[1,2] Mitrofanov[3] was the first to introduce the concept of GT to make the manufacturing systems more productive and efficient. CM or CMS is in fact the application of GT[4] in which parts are grouped into families and to fulfil the manufacturing requirements of each part family a dedicated group of machines is assigned to it.

CMS possesses the ability of producing medium variety and medium volume of parts economically as compared to other manufacturing systems.[5] The major advantages of CMS, already reported in literature, include reduced: setup time, Work-In-Process (WIP)[5–8], lead time[6], material handling costs[9], number of production equipment or tools needed[4,6], labour and overtime costs, and increased machine utilization[7] with faster response to internal and external changes such as: machine failures, product mix, and demand changes.[10]

The main challenge while implementing CMS is solving the Machine-Part Cell Formation (MPCF) problem which in fact is the grouping of parts into families and machines into corresponding machine groups.[11,12] MPCF can be handled in a stepwise manner: (i) formation of part families while considering similarities in design and processing requirements of parts, (ii) grouping of machines to process these part families, (iii) assigning each part family and its corresponding machine group to an individual cell.[13] These steps can be performed sequentially or simultaneously. The sequential strategy includes (i) Part-Family Identification (PFI) followed by (ii) Machine Groups Identification (MGI) and then parts are assigned to cells. Whereas, in simultaneous approach both PFI and MGI are carried out in parallel.[14]

The MPCF is a NP-hard problem[4,7] and that is why an extensive amount of literature has been dedicated to the solution of cell formation problem in the last more than five decades. Selim *et al*.[14] and Wemmerlov *et al*.[15] presented a comprehensive analysis of these techniques and their taxonomies. A more general distribution of the CMS design approaches include Classification and Coding (C&C)[16,17], Mathematical Programming[16,18], stochastic petri-nets[19], Heuristic Techniques[10,16] Meta-Heuristics and Hybrid Meta-Heuristics techniques.[16]

A considerable deal of attention has been focused on applying metaheuristic algorithms, which perform exceptionally well in combinatorial optimization problems, to a wide range of engineering challenges. Ghosh *et al*.[20] presented a review of different meta-heuristics presented in literature for handling the MPCF problem. Stawowy[21] developed an evolution-based technique using a modified permutation with a separator encoding method and a unique comprehension of the movement of separators during the mutation process. Wu *et al*.[22] developed a Water Flow-like Algorithm (WFA) to solve the MPCF problem, which mimicked the natural behavior of water flowing from higher to lower levels. Batsyn *et al*.[23] implemented a new pattern-based approach within the linear assignment model. The proposed approach consisted of two procedures, one for constructing an initial solution and another for achieving a final solution with higher Grouping Efficacy (GE). Karaboga and Basturk[24] presented a swarm-based approach utilizing Artificial Bee Colony (ABC). Nalluri *et al*.[25] developed a Hybrid Clonal Selection Algorithm (HCSA) that used a new affinity function in combination with a part assignment heuristic to handle the MPCF problem. Lei and Wu[26] utilized Tabu Search (TS) in combination with similarity coefficient to handle the cell formation problem. They also suggested a long-term TS algorithm to minimize the weighted sum of intercellular and intracellular movements and the total cell load variation.

Simulated annealing, both in standard and hybrid form, has also been reported to solve the MPCF problem.[27,28] Wu *et al*.[29] came up with an idea to optimize the initial clustering achieved via Similarity Coefficient Method (SCM) and Rank Order Clustering (ROC), the Boltzmann function of simulated annealing was combined with the mutation operator of the Genetic Algorithm (GA). Ateme-Nguema and Dao[30] developed an Ant algorithm-based TS heuristic for handling the CMS design problem. Solimanpur *et al*.[31] solved the MPCF problem using an Ant Colony Based optimization model.

In addition to the approaches reviewed so far, there is a larger chunk of literature that focused on utilizing GA, both in its standard and hybrid forms, to optimize the MPCF problem.[32–36] Some notable examples include: Goncalves & Resende[32], Tariq *et al*.[33], El Benani *et al*.[34], Javaid *et al.*[35] and more recently Salipour*et al al*.[36] Goli *et al*.[37] proposed a Fuzzy Mixed Integer Linear Programming (MILP), a Hybrid GA (HGA) and a Whale Optimization Algorithm (WOA) to estimate the role of Automatic Guided Vehicles (AGVs) in cell formation and parts scheduling. Maroof *et al*.[38] hybridized the standard GA for solving a multi-objective optimization problem to maximize the GE and Makespan (C_{max}) simultaneously.

Apart from GA, another tool that gained some attention, but has not been extensively implemented to handle the MPCF problem, is PSO.[39–43] Anvari *et al*.[39] proposed a hybrid PSO (HPSO)-based technique to handle the MPCF problem. This in fact was the first time a PSO based technique was applied to the CMS design problem. Another approach, employing PSO, was developed by Duran *et al.*[40]. In this approach the authors ignored the use of velocity vector as standard part of the algorithm and instead, the idea of proportional likelihood was used which is generally utilized in data mining techniques. Kao *et al.*[41] also developed a PSO-based technique with different process routings to minimize the exceptional parts movement outside the machine cells. Uthayakumar *et al.*[42] also came up with an algorithm utilizing PSO to minimize the intercellular movements and cell load variations. Mahmoodian *et al.*[43] proposed a combination of Artificial Intelligence (AI) with swarm intelligence while using Kohonen's learning rules. The results showed that the combination of algorithms has higher efficiency and efficacy than standard PSO algorithms.

To have a better view of the literature reviewed, all the references have been grouped technique/approach wise and presented in Table 1.

From the literature presented in Table 1, it can be evidently concluded that a considerable amount of literature is focused on the MPCF problem. A wide variety of techniques have been developed in the last more than fifty years. Keeping in view of the challenging nature of the problem, the focus of researchers has shifted from traditional approaches (C&C, Mathematical Programing, Heuristics) to Meta-Heuristics in the last couple of decades. However, even in Meta-Heuristics a larger chunk of research work, already available in literature, employs GA either in its standard or hybrid forms. On the other hand, a limited use of PSO gives the impression that GA might be comparatively more effective in handling the MPCF problem. Therefore, to investigate this aspect, a PSO based approach has been developed during this research and its results compared with some of the widely known approaches in general and GA, in particular. The comparisons showed that HPSO has been an efficient way to solve the MPCF problem as it outperformed the standard GA & PSO and remained equally efficient as far as the Hybrid GA is concerned.

Table 1 — A brief summary of the Literature review focused on the techniques reported for solving MPCF Problem

Approach[Ref]	Algorithms Classification
MathematicalProgramming[3,13,16,18,44,45]	Others
Heuristics[10,22,23,46,47]	
Clustering[48–50]	
Close Neighborhood Algorithm[51]	
Classification and Coding (C&C)[16,17]	
Tabu Search (TS)[26]	
Stochastic Petri Nets[19]	
Ant Colony Optimization[30,31]	
Hybrid Differential Evolution Algorithm (HGDE)[52]	
Simulated/Hybrid Simulated Annealing (SA/HSA)[27–29,53]	
Hybrid Clonal Selection Algorithm (HCSA)[25]	
Monarch Butterfly Optimization[7]	
GA/HGA[21,32–36,38,54]	GA – based
PSO/HPSO[4,39–43]	PSO – based

Methodology

The approach developed during this research is presented in Fig. 1. It has been coded in two stages using MATLAB. In the first stage a standard PSO is developed, and its results (35 benchmark problems) are compared with a standard GA – already reported in literature. In the second stage the same standard PSO is hybridized with a Local Search Heuristic (LSH) and its results are then compared with the standard PSO, developed in stage 1, and an HGA – already reported in literature.

Stage1 – Standard PSO Algorithm

The standard PSO technique is based on a randomly generated, initial population of possible solutions called particles which iteratively evolve towards the optimal solution by changing their initial positions with a certain velocity in accordance with the Eqs 1 & 2[(41)]

$$v_{ij}^{t} = \omega V_{ij}^{t-1} + c_1 r_1\left(p_{it}^{t-1} - x_{ij}^{t-1}\right) + c_2 r_2\left(g_j^{t-1} - x_{ij}^{t-1}\right) \quad \text{... (1)}$$

$$x_{ij}^{t} = x_{ij}^{t-1} + v_{ij}^{t} \quad \text{... (2)}$$

where, ω denotes the inertial weight, c_1 and c_2 are the self-adjustment and social adjustment learning factors, respectively, and r_1 & r_2 are the randomly selected values which lie between 0 and 1.

Hyperparameter Selection

Evolutionary computational algorithms, PSO being no exception, are sensitive to hyperparameters which, if not selected properly, may result in suboptimal performance and premature convergence. Extensive research has been conducted for the selection of hyperparameters.[55–57] In case of PSO Isiet *et al.*[55] experimented with varied values of these learning

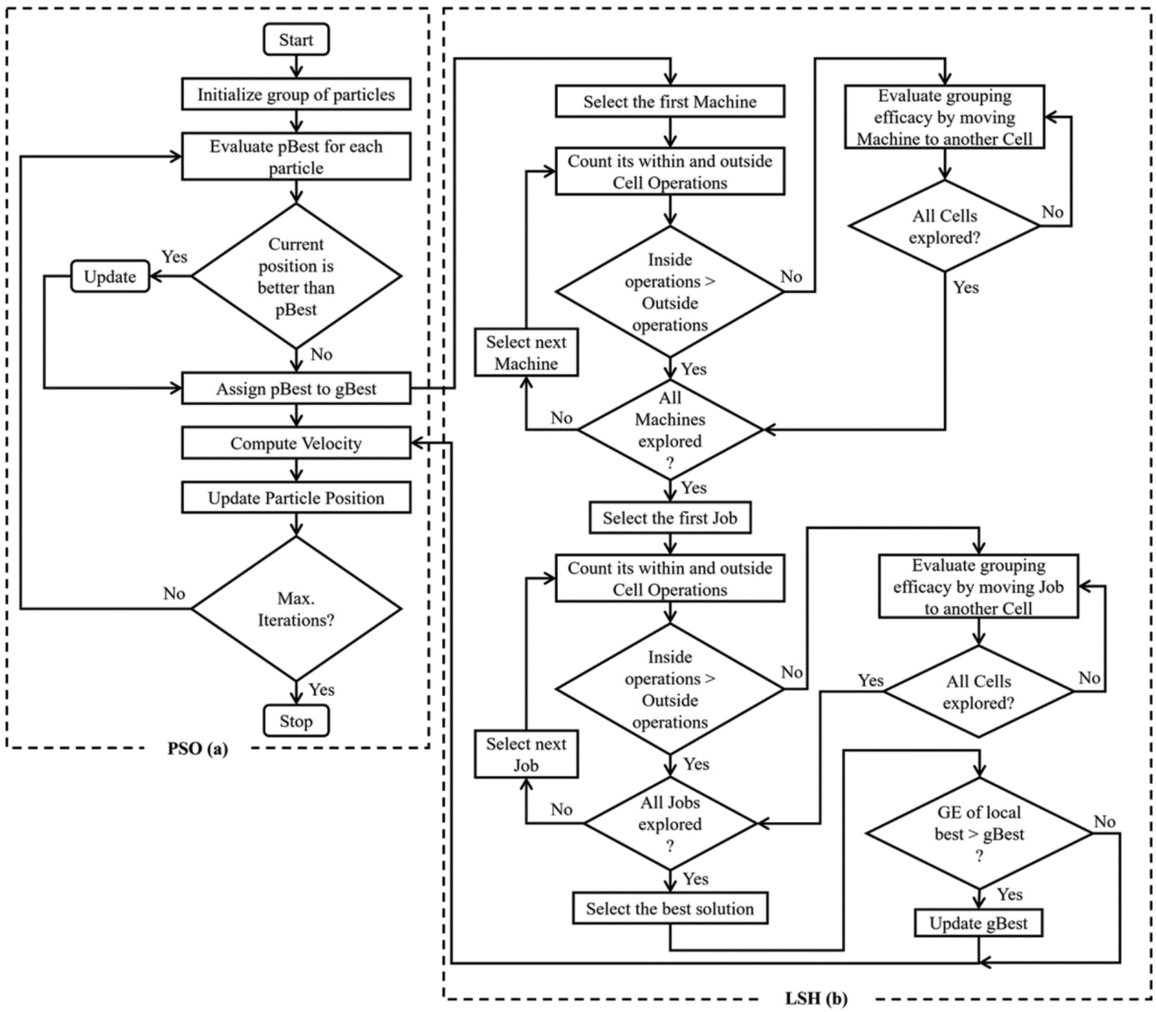

Fig. 1 — Methodology of the proposed HPSO

parameters. They finally suggested a range of [0 – 0.5] unlike the previously reported range of [0.9–1.2].(56,58) Zhu *et al.*[57] proposed that at the start of algorithm higher value of inertial constant be used for the purpose of enhanced exploration and be consistently reduced for reducing the effect of randomness in the algorithm and consequently helping it to converge to the optimum solution. Keeping the same approach in view, a range of [0.4–1.2] with a decrement factor of 0.975 has been used during this research. Furthermore, to keep the exploration aspect of the algorithm intact the learning factors C_1 (self-adjustment) and C_2 (Social-adjustment) has been kept at a value of 2.(59)

The step wise procedure adopted to implement the standard PSO algorithm is illustrated as follows:

a. Provide the input information about the machine-part incidence matrix and the number of cells
b. Randomly generate the initial population
c. Carry out the fitness evaluation of each particle i.e., determination of its Grouping Efficacy (GE)
d. Evaluate particle best (pBest) and global best (gBest)
e. Update the particle velocity and position to move towards gBest
f. Terminate if maxim iterations reached otherwise repeat the loop

For further clarity this stepwise procedure is also presented as a flow diagram in Fig. 1 (part a)

Stage2 – HPSO Algorithm

The standard PSO algorithm was evaluated for a set of standard benchmark problems (35 in total) widely reported in the literature. The results showed a significant performance degradation of the standard algorithm when dealing with large sized problems. To overcome this limitation, the algorithm was hybridized with a LSH resulting in an HPSO. The proposed LSH was applied to the gBest solution of the standard PSO algorithm obtained in each iteration, as illustrated in Fig. 1 (combination of part a & b). The integration of standard PSO with LSH is done in order to help the algorithm in converging to an optimum solution while avoiding the local optimums. The greedy nature of the LSH helps the algorithm to search in the neighborhood, of an already better solution found by the standard PSO, for its further improvement. The LSH, being an iterative search procedure, exhausts all possible combinations and retains improved version of the original solution. Implementation of stage 2 is expressed in a stepwise manner as follows:

a. Select the best solution generated by the standard PSO i.e., gBest.
b. Iteratively determine whether a machine/part is having more operations in its own cell than in other cells.
c. Improve the GE of machines/parts with more outside cell operations by iteratively moving the machine/part to other cells.
d. If the result of LSH i.e., local best (lBest) is better than gBest of PSO, then update the gBest with lBest and return to the main PSO loop.

The Machine-Parts incident matrix (MPIM), in binary form, is the main input to the algorithms. The layout of the matrix, as illustrated in Table 2, shows the parts and machines arranged along the rows and

Table 2 — Machine part incident matrix for the problem used for validation

	Machines																							
Parts	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0
2	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
3	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1
4	0	0	0	0	0	1	0	1	0	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0
5	0	0	0	0	0	1	0	1	0	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0
8	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
9	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0
10	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0
11	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
12	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
13	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0
14	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0
15	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
16	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0
17	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0
18	0	0	0	0	0	1	0	1	0	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0
19	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0
21	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
22	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0
23	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
24	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
25	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1
26	0	0	0	0	0	1	0	1	0	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0
27	0	0	0	0	0	1	0	1	0	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0
28	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
29	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0
30	0	0	0	0	0	1	0	1	0	0	0	1	0	0	1	0	0	1	0	0	0	0	0	0
31	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
32	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1
33	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0

(*Contd.*)

Table 2 — Machine part incident matrix for the problem used for validation (*Contd*.)

	Machines																							
34	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
35	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0
36	0	1	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0
37	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
38	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
39	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
40	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0

columns respectively. A value of '1' in a row against a particular column shows that the part is having an operation on that machine while a value of '0' means otherwise.

Performance Evaluation

There are several fitness criteria reported in the literature and used by researchers to determine the quality of a grouping solution. The two main measures that are frequently used are: Grouping Efficacy and Grouping Efficiency.[47] Grouping efficiency has been extensively used as a performance measure however, due to its lower capability of discrimination between good and bad solutions[32], Grouping Efficacy (GE) has been mostly preferred[60] and also utilized in this research. Pichandi *et al*.[47] specified the way GE is mathematically calculated as shown in Eq. 3

$$GE = (N_1 - N_1^{out})/(N_1 + N_0^{in}) \quad \ldots (3)$$

The parameter N_1 indicates the total number of 1s (operations) present in the matrix. N_1^{out} represents the total number of intercellular moves and N_0^{in} represents the number of zeros that are present inside the block-diagonal. The resulting value of GE ranges between 0 and 1 where 1 refers to no intercellular moves i.e., all parts are processed completely inside their assigned cells, whereas 0 means otherwise.

An initial validation of the approach developed during this research was carried out by solving a benchmark problem presented by Chandrashekharan and Rajagopalan[61] that has been extensively reported in the literature. The incident matrix (Table 2) is of size (24 x 40) i.e., it consists of twenty-four machines and forty parts. The result of the algorithm, which is the optimum block diagonalized form of the incident matrix, is shown in Table 3. It is evident from the results that the proposed LSH, which altered the best solution of the standard PSO by iteratively moving machines and/or parts from one cell to another to further improve its GE, resulted in an optimal solution (100 % GE) with a CPU time of 15.324 seconds.

Results and Discussion

To check the effectiveness of the proposed algorithms (PSO & HPSO) developed, a comparison is performed with GA and HGA, based on the results of 35 benchmark problems reported by Javaid *et al*.[35]. It has been observed that PSO remained effective in case of some small sized problems and consistently lost its effectiveness with the increase in problem size. On the other hand, the performance of HPSO remained consistent both in case of small and large sized problems as displayed in Table 4.

The PSO generated best results (max GE reported so far) for only 05 benchmark problems out of the total 35 (14.28%). However, its performance in comparison to GA remained marginally better as the later could only return 02 best solutions (5.71%). On the contrary the performance of their hybrid versions (HGA & HPSO) remained very impressive and consistent throughout, as both returned 28 (80%) and 29 (82.85%) best solutions. This huge performance boost is largely due to their hybrid nature that facilitates them to improve their ability of searching in the neighborhood of a better solution suggested by the standard version in each iteration. However, the performance of HPSO in comparison to HGA remained slightly better as it returned 29 best solutions i.e., 2.85 % more than HGA.

A comparison based on CPU time has also been carried out while using Intel Core 2 Duo 2.66 GHz, having 4 GB RAM. The comparison of the standard and hybrid methodologies based on CPU times for all the 35 benchmark problems, shown in Table 4, shows that PSO has been requiring the higher amount of CPU time in almost all the problems. The reason behind this is the standard nature of the tool due to which it requires larger number of iterations to reach a reasonable solution. However, despite consuming the larger amount of CPU time, its accuracy is just marginally better than GA. On the other hand, the HPSO's performance has been satisfactory in comparison to its counterpart (HGA). It has already been mentioned that in terms of accuracy HPSO has

Table 3 — Machine part incident matrix for problem validation

Parts	Machines																						
	1	13	21	22	3	4	16	20	2	5	11	19	7	14	23	24	6	8	12	15	18	9	10	17
1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
33	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
28	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
31	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
34	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
38	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
39	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
35	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
36	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
37	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Table 4 — Comparison of PSO & HPSO Developed During this Research with GA and HGA

Sr #	Source	Size	Num of Cells	GA[35]		HGA[35]		PSO		HPSO	
				GE	CPU time (S)	GE	CPU time (S)	GE	CPU time (S)	GE	CPU time (S)
1	[62]	5×7	2	82.35	0.396	82.35	0.47	82.35	0.346	82.35	0.147
2	[63]	5×7	2	60.87	0.201	69.57	0.29	69.57	0.460	69.57	0.422
3	[64]	5×18	2	62.50	0.647	80.85	0.64	79.59	1.058	80.85	0.801
4	[65]	6×8	2	76.92	0.508	79.17	0.45	76.92	0.503	79.17	0.390
5	[66]	7×11	3	48.48	0.760	60.87	0.77	58.62	0.951	60.87	0.595

(*Contd.*)

Table 4 — Comparison of PSO & HPSO Developed During this Research with GA and HGA (*Contd.*)

Sr #	Source	Size	Num of Cells	GA[35]		HGA[35]		PSO		HPSO	
				GE	CPU time (S)	GE	CPU time (S)	GE	CPU time (S)	GE	CPU time (S)
6	[53]	7×11	3	53.33	0.563	70.83	0.85	70.83	0.818	70.83	0.720
7	[64]	8×12	3	61.90	0.513	69.44	0.83	68.29	1.364	69.44	1.170
8	[61]	8×20	3	43.75	0.808	85.25	1.39	85.25	3.242	85.25	2.543
9	[61]	8×20	3	44.00	0.612	58.72	1.42	58.41	2.490	58.72	2.627
10	[67]	10×10	3	46.15	0.783	70.59	1.51	70.59	1.623	75.00	1.605
11	[49]	10×15	3	54.84	1.207	92.00	2.6	92.00	2.341	92.00	2.526
12	[68]	14×23	5	35.56	2.270	74.24	5.63	67.57	17.160	72.06	16.033
13	[69]	14×24	5	67.44	—	72.86	8.67	67.06	19.654	72.86	16.018
14	[70]	16×24	6	30.17	6.737	53.85	8.53	46.23	23.353	53.41	19.229
15	[71]	16×30	4	32.58	4.179	70.76	9.01	67.83	32.568	68.99	24.976
16	[48]	16×43	5	27.45	6.450	57.64	11.66	46.91	60.594	57.64	54.075
17	[72]	18×24	6	—	—	57.73	17.28	52.54	91.664	57.53	26.588
18	[73]	20×20	5	37.12	—	43.26	16.32	42.55	54.153	43.97	15.901
19	[74]	20×23	5	28.30	8.555	50.81	13.83	44.94	43.334	50.81	15.837
20	[72]	20×35	4	33.19	10.513	78.40	18.76	76.14	25.191	79.38	10.085
21	[51]	20×35	5	26.96	7.115	58.38	11.04	54.90	57.791	58.79	16.586
22	[75]	24×40	7	26.05	9.731	100	17.22	61.54	27.312	100.0	20.087
23	[75]	24×40	7	25.47	11.042	85.11	17.71	48.10	43.648	85.11	14.224
24	[75]	24×40	7	26.54	13.604	73.51	23.74	56.07	40.291	73.51	20.904
25	[75]	24×40	9	21.32	13.100	53.29	69.97	39.11	90.809	53.29	83.553
26	[75]	24×40	9	44.67	—	48.95	84.31	36.55	155.451	48.95	148.084
27	[75]	24×40	9	42.50	—	47.26	130.15	37.36	159.841	47.26	133.254
28	[70]	27×27	4	—	—	54.82	37.83	53.12	57.206	54.82	19.592
29	[72]	28×46	9	—	—	46.91	132.11	39.79	221.797	47.68	196.596
30	[76]	30×41	11	53.80	—	63.31	227.14	46.11	212.660	63.31	182.404
31	[69]	30×50	12	56.61	—	60.12	255.15	46.56	234.582	59.77	186.922
32	[69]	30×50	11	45.93	—	50.83	279.79	38.79	304.556	50.83	136.235
33	[62]	30×90	—	—	—	46.35	611.82	32.95	335.108	47.93	291.076
34	[70]	37×53	—	—	—	60.64	48.01	56.35	167.938	61.60	160.686
35	[75]	40×100	10	84.03	—	84.03	223.07	60.19	912.258	84.03	828.778
Total best solutions				02		28		05		29	

outperformed HGA by a small margin (2.85%). However, in terms of CPU time HGA has been having an upper hand by converging to best results in 22 benchmark problems quicker than the HPSO as the later (HPSO) has been faster in reaching the best solution in only 13 benchmark problems. Though this shows that HPSO is slower while converging to best solutions in comparison to HGA, however, by closely inspecting Table 3 it can be revealed that in the last 18 rows (problems from 18 to 35) the performance (in terms of CPU time) of HPSO and HGA is almost similar i.e., each tool has been converging to 9 best solution (50%). Furthermore, if CPU times of only the last 9 benchmark problems (problems from 27 to 35) are considered, then it can be observed that HPSO has been quicker in converging to the best solution in 5 problems (55%). This shows that as the problem size grows the performance of HPSO in terms of CPU time consumption has been getting better and better. It can be evidently concluded that the combination of PSO with LSH (HPSO) is comparatively more effective than its counterpart (HGA) and its true potential can be explored when tested against large sized problems (27 to 35).

A performance comparison of the HPSO, developed during this research, has also been carried out with several approaches from literature as shown in Table 5. The HPSO generates best results for 23 out of the total of 35 benchmark problems which is equivalent in terms of performance with HGA[35] and GA with Variable Neighborhood Search (GA-VNS)[77] and better than the rest especially the most recently reported approaches i.e. HGA[54] and CARIMO.[47]

Table 5 — Comparison of the proposed technique with other search algorithms

S/No	ZODIAC[46]	GRAFICS[50]	GATSP[78]	GA[79]	EA[32]	HGGA[80]	GAA[81]	EnGGA[82]	GA-LSH[33]	HGDE[52]	SA[27]	GA-VNS[77]	HGA[35]	HGA[54]	CARIMO[47]	This Approach (HPSO)
1	—	—	—	—	—	82.35	—	82.35	73.68	82.35	75	82.35	82.35	75	73.68	82.35
2	56.22	60.87	68	62.5	62.5	69.57	69.57	69.57	69.57	69.57	69.57	69.57	69.57	69.57	62.5	69.57
3	—	—	77.36	77.36	79.59	79.59	79.59	79.59	79.59	79.59	80.85	80.85	80.85	79.59	79.59	80.85
4	—	—	76.92	76.92	76.92	76.92	76.92	76.92	76.92	76.92	79.17	79.17	79.17	76.92	76.92	79.17
5	39.13	53.12	46.88	50	53.13	60.87	60.87	60.87	58.62	60.87	60	60.87	60.87	60.87	53.13	60.87
6	—	—	70.37	70.37	70.37	70.83	70.83	70.83	70.37	70.83	70.83	70.83	70.83	70.83	70.37	70.83
7	68.3	68.3	—	—	68.29	69.44	—	69.44	68.30	69.44	69.44	69.44	69.44	69.44	68.3	69.44
8	85.25	85.25	85.25	85.25	85.25	85.25	85.25	85.25	85.25	85.25	85.25	85.25	85.25	85.25	85.25	85.25
9	58.33	58.13	58.33	55.91	58.72	58.72	58.72	58.72	58.72	58.72	58.72	58.72	58.72	58.72	58.72	58.72
10	70.59	70.59	70.59	72.59	70.59	75	75	—	70.59	75	75	75	70.59	75	70.59	75
11	92	92	92	92	92	92	92	—	92	92	92	92	92	92	92	92
12	64.36	64.36	—	—	69.86	72.06	—	—	70.83	72.06	74.24	74.24	74.24	73.13	69.86	72.06
13	65.55	65.55	67.44	63.48	69.33	71.83	71.83	—	70.51	71.83	72.86	72.86	72.86	71.83	69.33	71.83
14	32.09	45.52	—	—	52.58	52.75	—	53.26	51.96	53.41	53.33	53.85	53.85	53.26	51.96	53.41
15	67.83	67.83	—	—	67.83	68.99	—	68.99	67.83	68.99	69.92	70.76	70.76	68.99	67.83	68.99
16	53.76	54.39	53.89	86.25	54.86	57.53	56.13	57.53	54.86	57.53	57.98	57.64	57.64	57.53	54.86	57.64
17	41.84	48.91	—	—	54.46	57.73	—	57.73	54.95	57.73	57.73	57.73	57.73	57.89	54.46	57.53
18	21.63	38.26	37.12	34.16	42.94	43.18	42.94	—	43.45	43.45	43.97	43.26	43.26	43.36	42.96	43.97
19	38.96	49.36	46.62	39.02	49.65	50.81	—	—	49.65	50.81	50.81	50.81	50.81	52.07	49.65	50.81
20	75.14	75.14	75.28	66.3	76.22	77.91	77.91	77.91	76.14	77.91	79.38	78.4	78.4	77.91	76.14	79.38
21	—	—	55.14	44.44	58.07	57.98	—	57.98	58.38	57.98	58.79	58.15	58.38	58.60	58.15	58.38
22	100	100	100	100	100	100	100	100	100.00	100	100	100	100	100	100	100.0
23	85.11	85.1	85.1	85.11	85.11	85.11	85.11	85.11	85.11	85.11	85.11	85.11	85.11	85.11	85.11	85.11
24	37.85	73.51	73.51	73.03	73.51	73.51	73.51	73.51	73.51	73.51	73.51	73.51	73.51	73.51	73.51	73.51
25	20.42	43.27	43.27	37.62	51.88	53.29	52.87	53.29	52.50	53.29	53.29	53.29	53.29	53.29	51.97	53.29
26	18.23	44.51	44.67	34.76	46.69	48.95	48.95	48.95	46.84	48.95	48.57	48.95	48.95	48.97	47.06	48.95
27	17.61	41.67	42.5	34.06	44.75	47.26	47.26	46.58	44.85	47.26	46	47.26	47.26	48.91	44.87	47.26
28	52.14	47.37	—	—	54.27	54.02	—	54.82	54.31	—	54.82	54.27	54.82	54.82	54.27	54.82
29	33.01	32.86	—	—	44.37	46.91	—	—	46.43	—	47.68	46.91	46.91	47.35	45.39	47.68
30	33.46	55.43	53.8	40.96	58.11	63.31	—	63.31	60.74	63.31	62.86	63.31	63.31	63.27	58.75	63.31
31	46.06	56.32	56.61	48.28	59.21	59.77	60.12	—	59.66	59.77	59.66	60.12	60.12	60.12	59.66	59.77
32	21.11	47.96	45.93	37.55	50.48	50.83	50.83	—	50.51	—	50.55	50.83	50.83	50.83	50.51	50.83
33	32.73	39.41	—	—	42.12	46.35	—	—	44.67	—	47.93	46.35	46.35	46.87	43.65	47.93
34	52.11	52.21	—	—	56.42	60.64	—	60.64	59.60	60.64	61.16	60.64	60.64	60.64	56.59	60.64
35	83.92	83.92	84.03	83.9	84.03	84.03	84.03	—	84.03	—	84.03	84.03	84.03	84.03	84.03	84.03
Total best sols	04	05	06	05	07	16	13	13	09	14	22	23	23	21	07	23

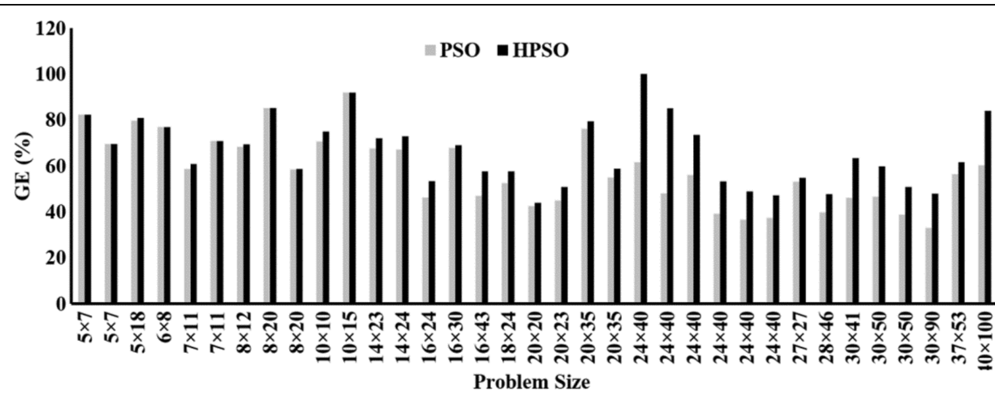

Fig. 2 — Comparison of PSO & HPSO for a set of 35 standard bench-mark problems

A graphical comparison of the GEs of the proposed HPSO algorithm with the standard PSO is shown in Fig. 2 for 35 benchmark problems. It is evident from the results that the proposed algorithm exhibits significantly better performance as compared to the PSO. The performance difference is more pronounced in large sized problems. A similar trend is also observed while comparing the CPU times of the two algorithms (Fig. 3). It is evident that the proposed algorithm is computationally more efficient as

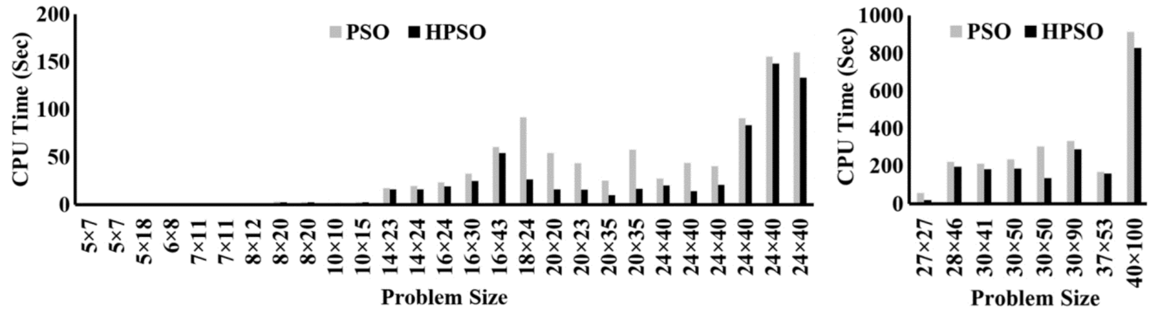

Fig. 3 — Comparison of computational time of both PSO & HPSO algorithms for a set of 35 standard bench-mark problems

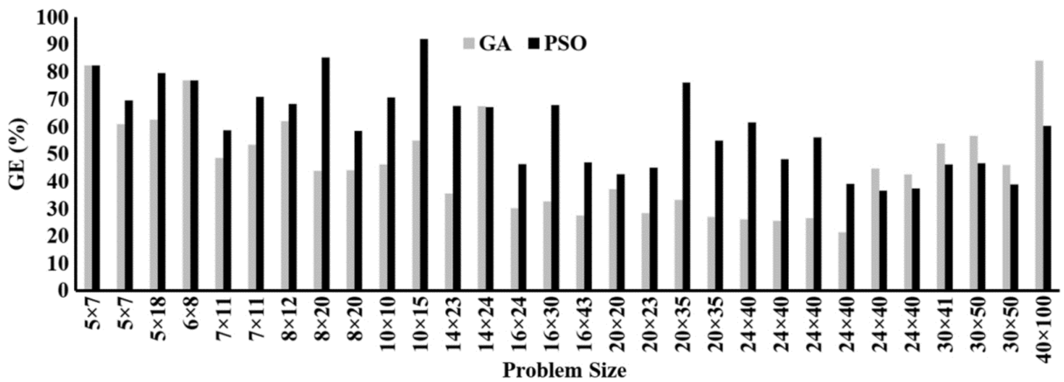

Fig. 4 — Performance comparison of GA and PSO based on GE. The GA performs better than the standalone PSO algorithm for larger problem sizes

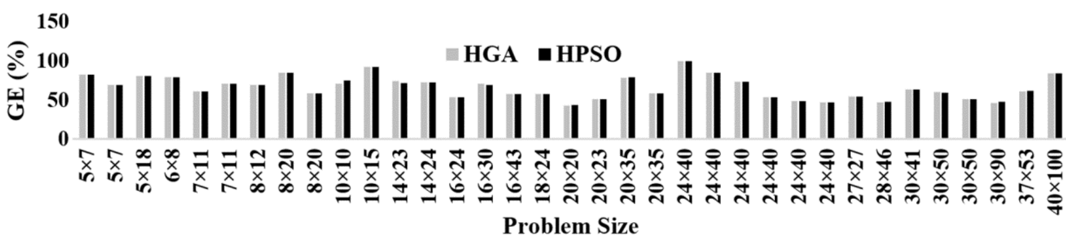

Fig. 5 — Performance comparison of HGA and HPSO based on GE

compared to the standard PSO. A particular reason for the performance difference in case of larger problems is the enhanced convergence ability of HPSO due to its integration with LSH. As the search space increases, because of the increased problem size, the PSO – that lacks the local searching capability – becomes comparatively less efficient. As a result, the CPU time for smaller sized problems is similar for both the algorithms, however a significant performance difference is evident in case of larger problems.

In addition to CPU time another comparison of the PSO with GA has been carried out on the basis of GE and presented in Fig. 4. In this case PSO's performance is significantly better in 24 benchmark problems. However, GA performs better in the last 5 problems which are comparatively larger in size. This indicates that the PSO, although computationally more efficient, loses efficacy when dealing with larger problems.

On the other hand, HPSO, which has been augmented with LSH, delivers a comparable performance with HGA as shown in Fig. 5. It is evident that the results for most of the problems are the same. Out of 35 benchmark problems, the HPSO generates the best results for 29 problems compared to 28 by HGA.

Conclusions

This research proposed an HPSO algorithm for effectively handling the MPCF problem. The results indicate an improved performance of the PSO algorithm hybridized with an LSH. The comparison with other evolutionary computational techniques shows the superior performance of the proposed algorithm especially against standalone PSO and GA. Although both the HPSO (proposed) & HGA algorithms exhibit a similar performance, the HPSO remained consistent in handling even large sized problems which shows that HPSO is comparatively robust in nature. One weakness of the proposed approach is its larger amount of CPU time consumption in case of small sized problems. This can be attributed to a comparatively larger amount of randomness induced in the algorithm while searching for the optimum and the iterative nature of the LSH. This research has proved the potential of PSO when hybridized with a LSH, however, in future further hybridization with other algorithms like Simulated Annealing can be tested for increased convergence. Also, the hybridized approaches may be tailored to handle multi-objective scenarios e.g. optimizing operational issues in addition to GE.

References

1 Aalaei A & Davoudpour H, A robust optimization model for cellular manufacturing system into supply chain management, *Int J Prod Econ*, **183** (2017) 667–679, https://doi.org/10.1016/j.ijpe.2016.01.014.

2 YounesSinaki R, Sadeghi A, Mosadegh H, Almasarwah N & Suer G, Cellular manufacturing design 1996–2021: A review and introduction to applications of industry 4.0, *Int J Prod Res*, **61**(**16**) (2023) 5585–5636, https://doi.org/10. 1080/ 00207543.2022.2105763.

3 Mitrofanov S P, Scientific principles of group technology (english translation) (National Lending Library for Science and Technology) 1966.

4 Husseinzadeh K A, Karimi B & Noktehdan A, A novel discrete particle swarm optimization algorithm for the manufacturing cell formation problem, *Int J Adv Manuf Technol*, **73**(**9–12**) (2014) 1543–1556, https://doi.org/ 10.1007/s00170-014-5906-4.

5 Hosseinabad E R, Adib M & Zaman U, A brief review on cellular manufacturing and group technology, *Res J Manag*, **5**(**1**) (2020) 1–20.

6 Ernawati D, Rahmawati N, Pudji E, Sari N K & Wianto A, Layout design in group technology using cellular manufacturing system, *J Phys Conf Ser*, **1569**(**3**) (2020) 032012, https://doi.org/10.1088/1742-6596/1569/3/032012.

7 Nagaraj G, Arunachalam M, Vinayagar K & Paramasamy S, Enhancing performance of cell formation problem using hybrid efficient swarm optimization, *Soft Comput*, **4** (2020) 16679–16690, https://doi.org/10.1007/s00500-020-05059-4.

8 Papaioannou G & Wilson J M, 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**) (2010) 509–521, https://doi.org/10.1016/j.ejor.2009.10.020.

9 Akturk M S & Turkcan A, Cellular manufacturing system design using a holonistic approach, *Int J Prod Res*, **38**(**10**) (2000) 2327–2347, https://doi.org/10.1080/00207 540050028124.

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

11 Jafarzadeh J, Amoozad Khalili H & Shoja N, A multiobjective optimization model for a dynamic and sustainable cellular manufacturing system under uncertainty, *Comput Intell Neurosci*, **2022** (2022) 1334081, https:// doi.org/10.1155/2022/1334081.

12 Renna P, Materi S & Ambrico M, Review of responsiveness and sustainable concepts in cellular manufacturing systems, *Appl Sci Switz*, **13**(**2**) (2023) 1125, https://doi.org/ 10.3390/app13021125.

13 Heragu SS & Chen J-S, Optimal solution of cellular manufacturing system design: Benders' decomposition approach, *Eur J Oper Res*, **107**(**1**) (1998) 175–192, https://doi.org/10.1016/S0377-2217(97)00256-7.

14 Selim H M, Askin R G & Vakharia A J, Cell formation in group technology: Review, evaluation and directions for future research, *Comput Ind Eng*, **34**(**1**) (1998) 3–20, https://doi.org/10.1016/s0360-8352(97)00147-2.

15 Wemmerlöv U & Hyer N L, Procedures for the part family/machine group identification problem in cellular manufacturing, *J Oper Manag*, **6**(**2**) (1986) 125–147, https://doi.org/10.1016/0272-6963(86)90021-5.

16 Arkat J, Hosseini L & Farahani M H, Minimization of exceptional elements and voids in the cell formation problem using a multi-objective genetic algorithm, *Expert Syst Appl*, **38**(**8**) (2011) 9597–9602, https://doi.org/10.1016/j.eswa. 2011.01.161.

17 Liao T W, Classification and coding approaches to part family formation under a fuzzy environment, *Fuzzy Sets Syst*, **122**(**3**) (2001) 425–441, https://doi.org/10.1016/S0165-0114(00)00033-6.

18 Plaquin M-F & Pierreval H, Cell formation using evolutionary algorithms with certain constraints, *Int J Prod Econ*, **64**(**1–3**) (2000) 267–278, https://doi.org/10.1016/ S0925-5273(99)00064-X.

19 Seo Y H, Kim T, Kim B N & Sheen D M, Representation and performance analysis of manufacturing cell based on generalized stochastic petri net, *Int J Ind Eng Theory Appl Pract*, **13**(**1**) (2006) 99–107, https://doi.org/10.23055/ ijietap.2006.13.1.427.

20 Ghosh T, Sengupta S, Chattopadhyay M & Dan P K, Meta-heuristics in cellular manufacturing: a state-of-the-art review, *Int J Ind Eng Comput*, **2(1)** (2011) 87–122, https://doi.org/10.5267/j.ijiec.2010.04.005.

21 Stawowy A, Evolutionary strategy for manufacturing cell design, *Omega*, **34**(**1**) (2006) 1–18, https://doi.org/10.1016/ j.omega.2004.07.016.

22 Wu T-H, Chung S-H & Chang C-C, A water flow-like algorithm for manufacturing cell formation problems, *Eur J Oper Res*, **205**(**2**) (2010) 346–360, https://doi.org/10.1016/ j.ejor.2010.01.020.

23 Batsyn M, Bychkov I, Goldengorin B, Pardalos P & Sukhov P, Pattern-based heuristic for the cell formation problem in group technology, in *Proc Mathemat Stat* (Springer New York LLC) 2013, 11–50, https://doi.org/10.1007/978-1-4614-5574-5_2.

24 Karaboga D & Basturk B, A powerful and efficient algorithm for numerical function optimization: Artificial bee colony

(abc) algorithm, *J Glob Optim*, **39**(**3**) (2007) 459–471, https://doi.org/10.1007/s10898-007-9149-x.

25 Nalluri M S R, Kannan K, Gao X Z & Roy D S, An efficient hybrid meta-heuristic approach for cell formation problem, *Soft Comput*, **23**(**19**) (2019) 9189–9213, https://doi.org/10.1007/s00500-019-03798-7.

26 Lei D & Wu Z, Tabu search approach based on a similarity coefficient for cell formation in generalized group technology, *Int J Prod Res*, **43**(**19**) (2005) 4035–4047, https://doi.org/10.1080/00207540500151283.

27 Pailla A, Trindade A, Parada V & Ochi L, A numerical comparison between simulated annealing and evolutionary approaches to the cell formation problem, *Expert Syst Appl*, **37**(**7**) (2010) 5476–5483, https://doi.org/10.1016/j.eswa.2010.02.064.

28 Wu T-H, Chang C-C & Chung S-H, A simulated annealing algorithm for manufacturing cell formation problems, *Expert Syst Appl*, **34**(**3**) (2008) 1609–1617, https://doi.org/10.1016/j.eswa.2007.01.012.

29 Wu T-H, Chung S-H & Chang C-C, Hybrid simulated annealing algorithm with mutation operator to the cell formation problem with alternative process routings, *Expert Syst Appl*, **36**(**2**) (2009) 3652–3661, https://doi.org/10.1016/j.eswa.2008.02.060.

30 Ateme-Nguema B H & Dao T M, Optimization of cellular manufacturing systems design using the hybrid approach based on the ant colony and tabu search techniques, in *IEEE Int Conf Ind Eng Eng Manag* (IEEE) 2007, 668–673, https://doi.org/10.1109/IEEM.2007.4419274.

31 Solimanpur M, Saeedi S & Mahdavi I, Solving cell formation problem in cellular manufacturing using ant-colony-based optimization, *Int J Adv Manuf Technol*, **50** (2010) 1135–1144, https://doi.org/10.1007/s00170-010-2587-5.

32 Gonçalves J F & Resende M G C, An evolutionary algorithm for manufacturing cell formation, *Comput Ind Eng*, **47**(**2–3**) (2004) 247–273, https://doi.org/10.1016/j.cie.2004.07.003.

33 Tariq A, Hussain I & Ghafoor A, A hybrid genetic algorithm for machine-part grouping, *Comput Ind Eng*, **56**(**1**) (2009) 347–356, https://doi.org/10.1016/J.CIE.2008.06.007.

34 Elbenani B, Ferland J A & Bellemare J, Genetic algorithm and large neighbourhood search to solve the cell formation problem, *Expert Syst Appl*, **39**(**3**) (2012) 2408–2414, https://doi.org/10.1016/j.eswa.2011.08.089.

35 Javaid W, Tariq A & Hussain I, A comparison of a standard genetic algorithm with a hybrid genetic algorithm applied to cell formation problem, *Adv Mech Eng*, **6** (2014) 301751, https://doi.org/10.1155/2014/301751.

36 Salimpour S, Pourvaziri H & Azab A, Semi-robust layout design for cellular manufacturing in a dynamic environment, *Comput Oper Res*, **133** (2021) 105367, https://doi.org/10.1016/j.cor.2021.105367.

37 Goli A, Tirkolaee E B & Aydin N S, Fuzzy integrated cell formation and production scheduling considering automated guided vehicles and human factors, *IEEE Trans Fuzzy Syst*, **29**(**12**) (2021) 3686–3695, https://doi.org/10.1109/TFUZZ.2021.3053838.

38 Maroof A, Tariq A & Noor S, An integrated approach for the operational design of a cellular manufacturing system, *Mehran Univ Res J Eng Technol*, **40**(**2**) (2021) 265–278, https://doi.org/10.22581/muet1982.2102.02.

39 Anvari M, Mehrabad M S & Barzinpour F, Machine–part cell formation using a hybrid particle swarm optimization, *Int J Adv Manuf Technol*, **47**(**5–8**) (2010) 745–754, https://doi.org/10.1007/s00170-009-2202-9.

40 Durán O, Rodriguez N & Consalter L A, A pso-based clustering algorithm for manufacturing cell design, in *Proc 1st Int ICST Conf Forensic Appl Techniq Telecommun, Inform Multimed* (ACM) 2008, https://doi.org/10.4108/wkdd.2008.2655.

41 Kao Y & Lin C-H, A pso-based approach to cell formation problems with alternative process routings, *Int J Prod Res*, **50**(**15**) (2012) 4075–4089, https://doi.org/10.1080/00207543.2011.590541.

42 Uthayakumar M, Adinarayanan A, Prabhakaran G, Slota A & Zajac J, Arrangement of machine cell in cellular manufacturing system: A pso approach, *J Adv Res Dyn Control Syst*, **10**(**5**) (2018) 247–257.

43 Mahmoodian V, Jabbarzadeh A, Rezazadeh H & Barzinpour F, A novel intelligent particle swarm optimization algorithm for solving cell formation problem, *Neural Comput Appl*, **31**(**S2**) (2019) 801–815, https://doi.org/10.1007/s00521-017-3020-x.

44 Krushinsky D & Goldengorin B, An exact model for cell formation in group technology, *Comput Manag Sci*, **9**(**3**) (2012) 323–338, https://doi.org/10.1007/s10287-012-0146-2.

45 Goldengorin B, Krushinsky D & Slomp J, Flexible pmp approach for large-size cell formation, *Oper Res*, **60**(**5**) (2012) 1157–1166, https://doi.org/10.1287/opre. 1120.1108.

46 Chandrasekharan M P & Rajagopalan R, Zodiac—an algorithm for concurrent formation of part-families and machine-cells, *Int J Prod Res*, **25**(**6**) (1987) 835–850, https://doi.org/10.1080/00207548708919880.

47 Pichandi R, Gupta N S & Rajendran C, CARIMO - a heuristic approach to machine-part cell formation, *Sadhana - Acad Proc Eng Sci*, **46**(**2**) (2021), https://doi.org/10.1007/s12046-021-01575-7.

48 King J R, Machine-component grouping in production flow analysis: an approach using a rank order clustering algorithm, *Int J Prod Res*, **18**(**2**) (1980) 213–232, https://doi.org/10.1080/00207548008919662.

49 Chan H M & Milner D A, Direct clustering algorithm for group formation in cellular manufacture, *J Manuf Syst*, **1**(**1**) (1982) 65–75, https://doi.org/10.1016/S0278-6125(82) 80068-X.

50 Srinivasan G & Narendran T T, GRAFICS—a nonhierarchical clustering algorithm for group technology, *Int J Prod Res*, **29**(**3**) (1991) 463–478, https://doi.org/10.1080/00207549108930083.

51 Boe W J & Cheng C H, A close neighbour algorithm for designing cellular manufacturing systems, *Int J Prod Res*, **29**(**10**) (1991) 2097–2116, https://doi.org/10.1080/00207549108948069.

52 Noktehdan A, Karimi B & Husseinzadeh K A, A differential evolution algorithm for the manufacturing cell formation problem using group based operators, *Expert Syst Appl*, **37**(**7**) (2010) 4822–4829, https://doi.org/10.1016/j.eswa.2009.12.033.

53 Boctor F F, A linear formulation of the machine-part cell formation problem, *Int J Prod Res*, **29**(**2**) (1991) 343–356, https://doi.org/10.1080/00207549108930075.

54 Hazarika M, An improved genetic algorithm for the machine-part cell formation problem, *Int J Syst Assur Eng Manag*, **14**(**1**) (2023) 206–219, https://doi.org/10.1007/s13198-021-01615-9.

55 Isiet M & Gadala M, Sensitivity analysis of control parameters in particle swarm optimization, *J Comput Sci*, **41** (2020) 101086, https://doi.org/10.1016/j.jocs.2020.101086.

56 Bansal J C, Singh P K, Saraswat M, Verma A, Jadon S S, *et al.*, Inertia weight strategies in particle swarm optimization, in *Third World Congress on Nature and Biolog Inspired Comput* (IEEE) 2011, 633–640, https://doi.org/10.1109/NaBIC.2011.6089659.

57 Zhu X & Wang H, A new inertia weight control strategy for particle swarm optimization, in *AIP Conf Proc*, **1955** (2018) 040095, https://doi.org/10.1063/1.5033759.

58 Shi Y & Eberhart R, Modified particle swarm optimizer, *Proc IEEE Conf Evol Comput* (IEEE) 1998, 69–73, https://doi.org/10.1109/ICEC.1998.699146.

59 Kennedy J & Eberhart R, Particle swarm optimization, in *Proc Int Conf Neural Netw*, (IEEE) 1995, 1942–1948, https://doi.org/10.1109/ICNN.1995.488968.

60 Noktehdan A, Seyedhosseini S & Saidi-Mehrabad M, A metaheuristic algorithm for the manufacturing cell formation problem based on grouping efficacy, *Int J Adv Manuf Technol*, **82**(**1**) (2015) 25–37, https://doi.org/10.1007/S00170-015-7052-Z.

61 Chandrasekharan M P & Rajagopalan R, An ideal seed non-hierarchical clustering algorithm for cellular manufacturing, *Int J Prod Res*, **24**(**2**) (1986) 451–463, https://doi.org/10.1080/00207548608919741.

62 King J R & Nakornchai V, Machine-component group formation in group technology: Review and extension, *Int J Prod Res*, **20**(**2**) (1982) 117–133, https://doi.org/10.1080/00207548208947754.

63 Waghodekar P H & Sahu S, Machine-component cell formation in group technology: Mace, *Int J Prod Res*, **22**(**6**) (1984) 937–948, https://doi.org/10.1080/00207548408942513.

64 Seifoddini H, A note on the similarity coefficient method and the problem of improper machine assignment in group technology applications, *Int J Prod Res*, **27**(**7**) (1989) 1161–1165, https://doi.org/10.1080/00207548908942614.

65 Kusiak A & Cho M, Similarity coefficient algorithms for solving the group technology problem, *Int J Prod Res*, **30**(**11**) (1992) 2633–2646, https://doi.org/10.1080/00207549208948181.

66 Kusiak A & Chow W S, Efficient solving of the group technology problem, *J Manuf Syst*, **6**(**2**) (1987) 117–124, https://doi.org/10.1016/0278-6125(87)90035-5.

67 Mosier C & Taube L, Weighted similarity measure heuristics for the group technology machine clustering problem, *Omega*, **13**(**6**) (1985) 577–579, https://doi.org/10.1016/0305-0483(85)90046-5.

68 Asktn R G & Subramantan S P, A cost-based heuristic for group technology configuration, *Int J Prod Res*, **25**(**1**) (1987) 101–113, https://doi.org/10.1080/00207548708919825.

69 Stanfel L E, Machine clustering for economic production, *Eng Costs Prod Econ*, **9**(**1–3**) (1985) 73–81, https://doi.org/10.1016/0167-188X(85)90012-6.

70 McCormick W T, Schweitzer P J & White T W, Problem decomposition and data reorganization by a clustering technique, *Oper Res*, **20**(**5**) (1972) 993–1009, https://doi.org/10.1287/opre.20.5.993.

71 Srinlvasan G, Narendran T T & Mahadevan B, An assignment model for the part-families problem in group technology, *Int J Prod Res*, **28**(**1**) (1990) 145–152, https://doi.org/10.1080/00207549008942689.

72 Carrie A S, Numerical taxonomy applied to group technology and plant layout, *Int J Prod Res*, **11**(**4**) (1973) 399–416, https://doi.org/10.1080/00207547308929988.

73 Mosier C & Taube L, The facets of group technology and their impacts on implementation-a state-of-the-art survey, *Omega*, **13**(**5**) (1985) 381–391, https://doi.org/10.1016/0305-0483(85)90066-0.

74 Kumar K, Kusiak A & Vannelli A, Grouping of parts and components in flexible manufacturing systems, *Eur J Oper Res*, **24**(**3**) (1986) 387–397, https://doi.org/10.1016/0377-2217(86)90032-9.

75 Chandrasekharan M P & Rajagopalan R, Groupabil1ty: An analysis of the properties of binary data matrices for group technology, *Int J Prod Res*, **27**(**6**) (1989) 1035–1052, https://doi.org/10.1080/00207548908942606.

76 Ravi K K, Kusiak A & Vannelli A, Grouping of parts and components in flexible manufacturing systems, *Eur J Oper Res*, **24**(**3**) (1986) 387–397, https://doi.org/10.1016/0377-2217(86)90032-9.

77 Paydar M M & Saidi-Mehrabad M, A hybrid genetic-variable neighborhood search algorithm for the cell formation problem based on grouping efficacy, *Comput Oper Res*, **40**(**4**) (2013) 980–990, https://doi.org/10.1016/J.COR.2012.10.016.

78 Cheng C H, Gupta Y P, Lee W H & Wong K F, A tsp-based heuristic for forming machine groups and part families, *Int J Prod Res*, **36**(**5**) (1998) 1325–1337, https://doi.org/10.1080/002075498193345.

79 Onwubolu G C & Mutingi M, A genetic algorithm approach to cellular manufacturing systems, *Comput Ind Eng*, **39**(**1–2**) (2001) 125–144, https://doi.org/10.1016/S0360-8352(00)00074-7.

80 James T L, Brown E C & Keeling K B, A hybrid grouping genetic algorithm for the cell formation problem, *Comput Oper Res*, **34**(**7**) (2007) 2059–2079, https://doi.org/10.1016/J.COR.2005.08.010.

81 Mahdavi I, Paydar M M, Solimanpur M & Heidarzade A, Genetic algorithm approach for solving a cell formation problem in cellular manufacturing, *Expert Syst Appl*, **36**(**3**) (2009) 6598–6604, https://doi.org/10.1016/J.ESWA.2008.07.054.

82 Tunnukij T & Hicks C, An enhanced grouping genetic algorithm for solving the cell formation problem, *Int J Prod Res*, **47**(**7**) (2009) 1989–2007, https://doi.org/10.1080/00207540701673457.