Int J Adv Manuf Technol (2016) 86:237–257
DOI 10.1007/s00170-015-8070-6

ORIGINAL ARTICLE

Applying soft-computing techniques in solving dynamic multi-objective layout problems in cellular manufacturing system

Tamal Ghosh[1] · **B. Doloi**[1] · **Pranab K. Dan**[2]

Received: 11 January 2013 / Accepted: 2 November 2015 / Published online: 7 December 2015

Abstract The formation of machine cells to process subsequent part families is not the only goal of the designing of an efficient cellular manufacturing system (CMS). A competent layout of the newly acquired cells is also essential to restrict the total inter-cell material handling cost which is primarily significant with large production volume. Furthermore, in realistic industrial scenario, the uncertainty of product demand can influence the layout configuration to be altered from period to period. Albeit there are numerous articles exist in the domain of CMS research considering cell formation problems, layout issues have not been addressed significantly. Therefore, the aim of our paper is to portray a reformed mathematical model of the inter-cell layout design problem in dynamic production situation considering material handling cost and a modified proximity relationship of manufacturing cells. The proposed Quadratic Assignment Programming (QAP) model is combinatorial in nature and is difficult to solve using traditional exact solution methods. The state-of-the-art soft-computing techniques are extremely advantageous for such QAP paradigms. Thus, we developed an improved genetic algorithm (IGA) and a simulated annealing heuristic (SAH) to sort out the abovementioned problem. Due to the inadequacy of datasets, we formed small to large size test problems (6×6×2 to 24×24×10) in logical way to cater the purpose. The proposed algorithms are successfully employed to attain near-optimal solutions to the test problems. Computational results demonstrate the proficiency of the IGA over SAH for all the test problems in terms of solution quality and computational time. In addition, we conducted a statistical data analysis to validate the test results.

Keywords Multi-objective cellular layout · Production uncertainty · Material handling · Proximity factor · Inter-cell material flow · Genetic algorithm · Simulated annealing

1 Introduction

Recently, manufacturing and service enterprises are being operated more commendably with reduced throughput time due to intense competitive market and demand uncertainty for the end products. Therefore, manufacturing firms are being forced to invest less capital for the important resources such as raw materials, assets, and labors to reduce the overall production costs and shorten the production lead time [1]. Among the latest manufacturing philosophies, group technology (GT) has been successfully applied in an way to reduce the throughput time and material handling cost by dominating the work in progress and finished goods inventories and improving the competence to deal with the forecast inaccuracies in uncertain production conditions [2, 3]. Cellular manufacturing (CM) is a function of GT and has been evolved as a potential replacement of traditional manufacturing systems. CM could be exemplified as a hybrid system which exploits the advantages of job shop (variety) and flow shop (high rate of production) production strategies. The steps of designing an efficient CMS are (1) to decompose the manufacturing system into cells by recognizing and exploiting the similarities among

✉ Tamal Ghosh
tamal.31@gmail.com

B. Doloi
bdoloionline@rediffmail.com

Pranab K. Dan
danpk.wbut@gmail.com

[1] Production Engineering Department, Jadavpur University, 188, Raja S. C. Mullick Road, Kolkata 700032, India

[2] Rajendra Mishra School of Engineering Entrepreneurship, IIT Kharagpur, Kharagpur 721302, India

components and machineries and (2) to design efficient inter-cell and intra-cell layout in order to ease the material flow on shop-floor. The first step is identified as the classical cell formation problem (CFP) which forms machine cells to process the corresponding part families completely in order to avoid the inter-cell part travels. However, this concept might not be feasible and could be inexpensive in real production scenario since the cells are not practically independent to each other [4]. If some of the operations of a part are to be performed beyond its granted cell, then that part is recognized as an "exceptional element," and the machines performing those tasks and do not belong to its dedicated cell are distinguished as "bottleneck" machines. The newly formed cells are believed to be assigned to optimal locations inside the factory area to control the inter-cell material flow being caused by the presence of bottleneck machines. This problem is categorized as cell layout problems (CLP) [5], identified as the second step of an efficient CMS design. A competent layout in CMS not only improves its performance but also minimizes nearly 40–50 % of the total production cost [6]. However, layout design in CMS has not gain much attention of researchers in recent past since most of the CMS researchers are somehow working on the cell formation issues [7]. The layout problem in CMS is classified as the classical Quadratic Assignment Programming (QAP) problem and is believed to be NP-complete in nature [8]. To realize the essence of CLP, we classified the literature of layout problems as follows:

- *Non-cellular layout problems*
- *Cellular layout problems*

1.1 Non-cellular layout problems

This subsection demonstrates the distinct issues of static and dynamic layout problems. The static layout problems (SLP) are also identified as single-period layout problems. An SLP could be realized either using qualitative or a quantitative methodologies. These techniques are often used singly or jointly. Fortenberry and Cox [9], Urban [10], and Harmonosky and Tothero [11] previously discussed the need of combined objectives in terms of closeness rating as qualitative factor and material handling cost as quantitative factor for SLP.

In varying demand scenario, uncertainty is supposed to be adopted in the layout model formulation to obtain efficient layout for each period over the entire planning horizon. It simply means a layout constructed in a planning period might not be efficient for the next period. Therefore, a dynamic version of the layout problem is required which is an extension of the SLP [12–15]. The dynamic layout problem (DLP) is more complex since the stochastic demand of end products would complicate the forecast decision of any manufacturing firm. The average lifetime of a layout is approximately 3 years for a particular period. Lacksonen and Enscore demonstrated a DLP considering the material flow cost and department rearrangement cost over multiple planning periods and employed a cutting plane method to solve one of the largest problems (30×30×5) of DLP literature efficiently [16]. Kaku and Mazzola proposed a proficient tabu search technique that exploits aspiration criteria, dynamic tabu list strategies, and other strategies for search intensifications and diversifications to solve the DLP [17]. They proposed the following DLP model:

$$\text{Minimise } Z = \sum_{p=1}^{T}\left[\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{N}\sum_{l=1}^{N} \mathrm{mfr}_{ikp}\,\mathrm{distn}_{jl}\,c_{ijp}c_{klp} + \sum_{i=1}^{N}\sum_{j=1}^{N} h_{ijp}c_{ijp} + \sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{l=1}^{N} vr_{ip}b_{ip} + fr_p a_p\right] \quad (1)$$

Subject to:

$$\sum_{i=1}^{N} c_{ijp} = 1 \quad j\in N \quad (2)$$

$$\sum_{j=1}^{N} c_{ijp} = 1 \quad i\in N \quad (3)$$

$$c_{ijp} = \begin{cases} 1, & \text{if cell } i \text{ is assigned to location } j \text{ in period } p \\ 0, & \text{Otherwise} \end{cases} \quad (4)$$

$$b_{ip} = \begin{cases} 1, & \text{if cell } i \text{ is moved at the beginning of period } p \\ 0, & \text{Otherwise} \end{cases} \quad (5)$$

$$a_p = \begin{cases} 1, & \text{if any change is made at the beginning of period } p, \\ 0, & \text{Otherwise} \end{cases} \quad (6)$$

$$b_{ip} = \sum_{j=1}^{N} c_{ijp-1}\left(\sum_{l=1}^{N} c_{ilp}\right) \quad i\in N,\ l\in N\backslash\{j\},\ p\in T\backslash\{1\} \quad (7)$$

$$a_p \geq b_{ip} \quad i\in N,\ p\in T\backslash\{1\} \quad (8)$$

$$b_{ip},\ a_p \geq 0 \quad i\in N,\ p\in T\backslash\{1\} \quad (9)$$

where

- distn_{jl} distance between two locations j and l (j, l=1, 2, 3…, N)
- mf_{ikp} the amount of material flow from cell i to k in period p (i, k=1, 2, 3, …, N; p=1, 2,3, …, T, $\mathrm{mf}_{ikp}=\mathrm{mf}_{kip}$, $i\neq k$)
- mfr_{ikp} the inter-cell trips between cell i and k in period p ($\mathrm{mfr}_{ikp}=\mathrm{mf}_{ikp}+\mathrm{mf}_{kip}$) (for i, k=1, 2, 3,…, N; p=1, 2, 3,…, T)
- h_{ijp} the cost of assigning cell i to location j in period p
- vr_{ip} variable reconfiguration cost of moving cell i at the beginning of period p

fr_p fixed reconfiguration cost related with making any alterations in layout at the beginning of period p
c_{ijp} the decision variable
b_{ip} the decision variable

Urban employed the concept of incomplete dynamic programming to the DLP to find the optimal solution to the problem with fixed rearrangement costs with extremely reduced CPU time [18]. Sahin et al. stated that the solution to the DLP has twofold meaning; the key goals are to minimize the material flow cost of each planning period and the rearrangement cost of moving facilities from period to period [19]. In general, a DLP of n-cells and p-periods can produce $(n!)^p$ solutions, which means it is required to investigate 6.3587×10^{86} layouts for a $12\times12\times10$ problem to find a near-optimal solution. It is a huge computational task to be performed even for a modern computer. Due to this reason, the classical optimization methods or exact algorithms are not competent in obtaining good solutions for such problems. A trend has been identified of using heuristic approaches that can efficiently solve such problems and attain near-optimal solutions at the expense of a moderate computational effort [15, 19, 20].

1.2 Cellular layout problems

Logendran first proposed a mathematical model that considered the sequence of operations in evaluating the inter-cell and intra-cell moves and the impact of the cell layout to illustrate the inter-cell material flow [21]. Alfa et al. suggested a concurrent sub-optimal solution of the machine grouping and layout problem in CMS using a simulated annealing (SA) approach [22]. Sarker and Yu reported a twofold procedure for duplicating bottleneck machines in CMS and solved a cell layout problem which minimizes the total inter-cell material flow [23]. A flow-network-oriented inter-cell layout design model is defined by Tang and Abdel-Malek in three key steps: (1) K shortest path method to reduce various flows of a system into a master flow network; (2) a flow pattern which designates the system's aisle structure; (3) cell allocation around the flow pattern and the aisle structure within a limited area floor plan [24]. Lee adopted the intra-cell and inter-cell layout design problems in his model using three-phase interactive method following the decomposition strategy to reduce the large problem into smaller sub-problems with minute details [25]. Salum introduced some similarity measures to construct an intra-cell layout by placing machines with higher similarity value next to each other to minimize the total material handling time in the system [26]. Wang and Sarker stated a lower bound on the inter-cell layout problem (QAP) and prescribed a 3-pair comparison heuristic and "bubble search" technique-based layout design algorithms in order to minimize the inter-cell material flow incurred due to bottleneck machines [7]. Authors adopted the following QAP model:

$$\text{Minimise } Z=\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{N}\sum_{l=1}^{N} \mathrm{mfr}_{ik}\,\mathrm{distn}_{jl}c_{ij}c_{kl} \tag{10}$$

subject to:

$$\sum_{i=1}^{N} c_{ij}=1 \quad j\in N \tag{11}$$

$$\sum_{j=1}^{N} c_{ij}=1 \quad i\in N \tag{12}$$

$$c_{ij}=\begin{cases}1, & \text{if cell } i \text{ is assigned to location } j\\ 0, & \text{Otherwise}\end{cases} \tag{13}$$

Chan et al. proposed Main algorithm to solve intra-cell layout problems in static and dynamic conditions by considering part-handling factor, machine rearrangement cost, and machine closeness factor [27]. Solimanpur et al. developed an ant colony algorithm to solve the QAP model proposed in reference [7] and compared their results with other techniques such as H63, HC63-66, CRAFT, and bubble search with improved solutions [28]. Wu et al. [29] successfully implemented a new genetic algorithm (GA) to solve the cell design and group layout problems concurrently incorporating some important production factors, such as operational sequences, part demand, transfer batch, machine capacities, and layout types. Reference [30] proposed a GA-based algorithm to solve cell layout-based QAP along with a cell formation model considering the linear layout shape. Kulkarni and Shankar [5] employed a GA to the inter-cell layout problem stated in references [7, 28] and validated the performance of their algorithm with well-known layout design techniques.

Tavakkoli-Moghaddam et al. [31] presented a new mathematical model to solve a layout problem with varying demand to minimize the inter-cell and intra-cell layout costs simultaneously. Mahdavi and Mahadevan developed an algorithm that concurrently obtains manufacturing cells and the intra-cell layout successfully [32]. Ahi et al. applied a TOPSIS-based initial solution-generating method for order preference by similarity to the ideal solution that leads to determination of cell formation, intra-cell, and inter-cell layouts. Authors have shown further improvements to the proposed method [33]. Ariafar and Ismail proposed a new QAP model for inter-cell and intra-cell layouts and solved using an SA algorithm with optimal solution [34]. Ma and Zhang demonstrated that the dynamic layout framework based on reconfigurable CMS aiming at the enterprise problems is based on alternative

process routes and multiple machine types available for the operations considering cell formation and inter-cell layout jointly [35]. Jolai et al. employed a binary PSO-based new heuristic approach to solve a QAP model for inter-cell and intra-cell layout problems considering uncertain demand of parts and batch sizes using a variable neighborhood search [36].

Leno et al. recently discussed the multi-objective model for cellular layout design problems. It minimizes the total material handling cost in the first place and subsequently maximizes the distance-weighted closeness factor of cells and solved that model using a GA with near-optimal solutions [37]. Arkat et al. [38] employed two techniques based on GA to solve an integrated model of cell formation, cell layout, and cell scheduling. The method is capable of achieving near-optimal solutions. Similar issues are addressed by Kia et al. by developing a novel nonlinear programming model and solved using an SA algorithm and compared successfully with the solutions of Lingo software [39].

We have summarized the outcomes of this brief literature review in Table 1.

The following facts can be stated henceforth:

a) Most of the articles considered a combined form of layout and cell formation problems which may perhaps complicate the model unnecessarily while the cell layout problem could be addressed independently in more efficient way.
b) Few researchers have proposed the QAP model as a combined form of inter-cell and intra-cell layout problems. However, the inter-cell and intra-cell layout subject matters might be discussed individually due to the consequence of critical and intrinsic factors related to CMS, such as product mix, demand variation, lot sizing, machine similarity, cell adjacency, and machine utilization.
c) Most of the researchers considered single-period layout model which is not very realistic in practical scenario due to the product mix and varying demand of end products. Static model does not consider re-layout cost which is an essential cost component in realistic production scenario.
d) Single- and multi-period layout problems are not distinctly and prominently discussed in the CMS literature as it is done in the domain of generalized layout problems.

To address these issues, an adapted multi-objective mathematical model is demonstrated in this article which formulates the inter-cell layout problem efficiently. Adjacency and separation factors are resolved in this QAP model in terms of "relative proximity factor." Since this model is proposed for the multi-period planning horizon, therefore reconfiguration cost of layout from period to period is also incorporated. The cell formation solution, cell-to-cell and location-to-location distances, material flow among cells, variable and fixed reconfiguration costs, and assignment costs of cells to locations are the inputs to the model. Due to the combinatorial nature of the model, a simulated annealing heuristic (SAH) technique and an improved genetic algorithm (IGA) are developed to obtain the near-optimal solutions for this problem. The rest of the article is structured in the following manner. In Section 2, we introduce the formulation of the problem as a QAP model; in Section 3, we illustrate the proposed SAH and IGA algorithms. The simulation study and results are conferred in Section 4. Lastly, in Section 5, we conclude our research and show the proper direction of the possible future extension of this study.

2 Problem formulation

The layout problem in CMS is a Quadratic Assignment Problem (QAP) which restrains each cell to be assigned to only one location and each location to be selected for only one cell. The multi-period cellular layout problem (MPCLP) can be stated as an expansion of single-period cellular layout problem (SPCLP). To define the generality of MPCLP, the best layout can be selected for each period and then to decide whether to change it in next period which further incurs the reconfiguration or re-layout cost. If reconfiguration cost is trivial, the problem would become the sequence of SPCLPs. However, in experience, reconfiguration incurs the overall costs. Therefore, the layout for each period must influence the layout for subsequent period because the overall cost is computed over the entire planning horizon.

The suppositions of the proposed model are as follows:

- Manufacturing cells are already developed and cell formation solutions are available.
- The size of each cell is equal, and the shape and space of the floor area are not limited.
- The distance between two locations is the measurement from one center to another.
- The material flow from one cell to another is measured beforehand.

In the domain of CMS, past literature rarely presents any multi-objective cellular layout model which practically formulates quantitative and qualitative objectives excepting reference [37] which proposed a multi-objective model for SPCLP recently. However, multi-objective MPCLP has not yet been proposed in CMS research.

We introduce a multi-objective MPCLP model considering both the qualitative (relative proximity) and quantitative (material flow) objectives. Due to the dynamic nature of product demand, different manufacturing firms

Table 1 Summary of the literature review done on cellular layout problems

References	Nature of the layout problem		Scope of the layout problem considered		Objectives considered (either minimization type (total cost) or maximization type (profit/closeness/similarity/machine utilization))
	Static	Dynamic	Inter-cell	Intra-cell	
Logendran 1991 [21]	×		×	×	Total inter-cell and intra-cell moves and utilization of workstations
Sarker and Yu 1994 [23]	×		×		Inter-cell material flow and bottleneck machine that need to be duplicated
Tang and Abdel-Malek 1996 [24]	×		×		Cell-to-cell material flow considering a flow pattern within strict floor plan
Lee 1998 [25]	×		×	×	Total inter-cell and intra-cell material handling cost
Salum 2000 [26]	×			×	Total manufacturing lead time (MLT) reduction and intra-cell flow
Wang and Sarker 2002 [7]	×		×		Total inter-cell material flow
Alfa et al. 1992 [22]	×		×	×	Total flow of material (combined form of cell formation and layout design)
Chan et al. 2004 [27]	×	×		×	Material handling cost and rearrangement cost
Solimanpur et al. 2004 [28]	×		×		Total inter-cell material flow
Wu et al. 2006 [29]	×		×	×	Total flow of material (combined form of cell formation and layout design)
Chan et al. 2006 [30]	×		×		Total inter-cell material flow (combined form of cell formation and layout design)
Kulkarni and Shankar 2007 [5]	×		×		Total inter-cell material flow
Tavakkoli-Moghaddam et al. 2007 [31]	×		×	×	Total flow of material
Mahdavi and Mahadevan 2008 [32]	×			×	Total intra-cell material flow (combined form of cell formation and layout design)
Ahi et al. 2009 [33]	×		×	×	Total flow of material (combined form of cell formation and layout design)
Ariafar and Ismail 2009 [34]	×		×	×	Total inter-cell and intra-cell material handling cost
Ma and Zhang 2010 [35]		×	×	×	Total flow of material (combined form of cell formation and layout design)
Jolai et al. 2011 [36]	×		×	×	Total inter-cell and intra-cell material handling cost
Arkat et al. 2011 [38]	×		×	×	Total flow of material (combined form of cell formation and layout design considering cell scheduling decisions)
Leno et al. 2011 [37]	×		×		Total flow of material, total distance-weighted closeness rating and penalty to force the solutions to satisfy floor boundary condition
Kia et al. 2012 [39]		×		×	Total intra-cell material flow (combined form of cell formation and layout design considering cell scheduling decisions)

consider several different aspects related to the process of production. Shared utilities of different cells could be incorporated along with the susceptible environmental factors. We adopted three simple approaches as stated in reference [11]:

- Quantify the qualitative goals in order to analyze the effect mathematically
- In order to institute equal impacts, normalize all the factors in the layout design
- Assign load factor to the qualitative objective to demonstrate its relative magnitude with respect to the material flow in the final layout design

We defined a novel relationship among cells which is qualitative in nature and it combines two important qualitative relationships, adjacency-based relationship (commonly used in layout design literature [9, 11, 40]) and distance- or separation-based qualitative relationship (generally considered in order to place some cells completely separated from

Table 2 An adjacency and separation relationship matrices of the cells (6×6 dataset)

ADJ	C1	C2	C3	C4	C5	C6	SEP	C1	C2	C3	C4	C5	C6
C1	–	E	O	U	A	I	C1	–	ζ	α	β	ε	δ
C2		–	E	U	A	U	C2		–	δ	ε	γ	β
C3			–	X	U	X	C3			–	γ	β	α
C4				–	U	U	C4				–	α	δ
C5					–	A	C5					–	γ
C6						–	C6						–

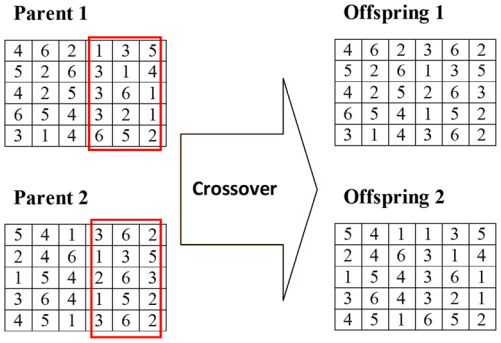

Fig. 1 Crossover operation for (6×6×5) test problem

some others due to environmental disputes, such as noise, vibration, safety, pollution, and other risk related to fire or detonation). The combined form of these two issues is termed as *relative proximity factor*. This relationship could be defined as follows:

$$\text{prox}_{ikp} = \text{ajdc}_{ikp} + \text{sep}_{ikp}$$

This fact could be realized using some suitable example. For that matter, two example test problems are considered in the Table 2. Since these relationships are qualitative in nature, Table 2 reflects qualitative values of the relationships. These different qualitative relationships are illustrated using numerical scoring systems.

For example, the adjacency relationships can be demonstrated by the following numerical values: $A=4$, $E=3$, $I=2$, $O=1$, $U=0$ and $X=-1$ obtained from reference [40].

Further, the separation relationships can be articulated using the following numerical values: $\alpha=9$, $\beta=7$, $\gamma=5$, $\delta=3$, $\varepsilon=1$, $\zeta=-1$. The physical meaning of the separation relationships of Table 2 is as follows:

α= furthest separation, β= strong separation, γ= moderate separation, δ= weak separation, ε= weakest separation, ζ= separation not desirable.

The numerical values of these two qualitative factors can be combined as stated above (e.g., $\text{adjc}_{12p}+\text{sep}_{12p}=E+\zeta=3+(-1)=2=\text{prox}_{12p}$) and the "relative proximity factor" matrix is obtained in Table 3.

The weighted QAP formulation of MPCLP proposed in this research is the combined form of two objectives as ($Z1+C\times Z2$) as suggested by Urban [10]:

$$\text{Minimize } Z=\sum_{p=1}^{T}\left[\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{k=1}^{N}\sum_{l=1}^{N}\left(N\text{m}\,\text{fr}_{ikp}+w\times N\text{prox}_{ikp}\right)\times \text{distn}_{jl}c_{ijp}c_{klp}+\sum_{i=1}^{N}\sum_{j=1}^{N}h_{ijp}c_{ijp}+\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{l=1}^{N}\text{vr}_{ip}b_{ijlp}+\text{fr}_{p}a_{p}\right] \quad (14)$$

subject to:

$$\sum_{i=1}^{N} c_{ijp}=1 \quad j\in N \quad (15)$$

$$\sum_{j=1}^{N} c_{ijp}=1 \quad i\in N \quad (16)$$

Table 3 Relative proximity factor matrix of the cells (6 cells×6 locations problem)

	C1	C2	C3	C4	C5	C6
C1	–	2	4	7	5	5
C2		–	6	1	9	7
C3			–	4	7	8
C4				–	9	3
C5					–	9
C6						–

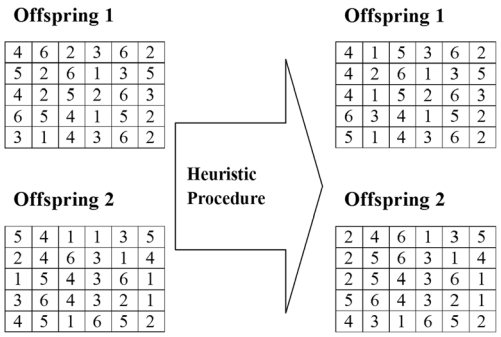

Fig. 2 Restructure of offsprings

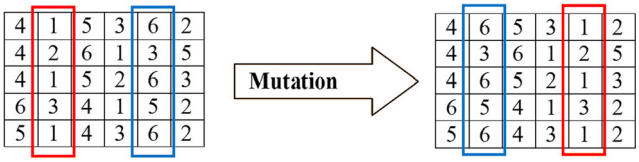

Fig. 3 Mutation operation for a chromosome of (6×6×5) test problem

$$b_{ijlp} = \sum_{j=1}^{N} c_{ijp-1}\left(\sum_{l=1}^{N} c_{ilp}\right) \quad i\in N,\ l\in N\backslash\{j\},\ p\in T\backslash\{1\} \tag{17}$$

$$a_p \geq b_{ip} \quad i\in N,\ p\in T\backslash\{1\} \tag{18}$$

$$b_{ijlp},\ a_p \geq 0 \quad i\in N,\ p\in T\backslash\{1\} \tag{19}$$

$$c_{ijp} = \begin{cases} 1, & \text{if cell i is assigned to location } l \text{ in period } p \\ 0, & \text{Otherwise} \end{cases} \tag{20}$$

$$b_{ijlp} = \begin{cases} 1, & \text{if cell } i \text{ is moved from location } j \text{ to } l \text{ at the beginning of period } p \\ 0, & \text{Otherwise} \end{cases} \tag{21}$$

$$a_p = \begin{cases} 1, & \text{if any change is made at the beginning of period } p, \\ 0, & \text{Otherwise} \end{cases} \tag{22}$$

where

b_{ijlp}	the decision variable
adjc_{ikp}	adjacency rating between cells i and k for the period p
sep_{ikp}	separation rating between cells i and k for the period p
prox_{ikp}	relative proximity factor between cells i and k for the period p
S_{ik}	the relationship value between cells i and k
T_{ik}	the normalized relationship value between cells i and k
$N\text{mfr}_{ikp}$	normalized value of mfr_{ikp}
$N\text{prox}_{ikp}$	normalized value of prox_{ikp}

As soon as all the relationship factors are quantified, a method of normalization of data is applied using the following formula adopted from reference [11]:

$$T_{ik} = \frac{S_{ik}}{\sum_{i=1}^{N}\sum_{k=1}^{N} S_{ik}} \tag{23}$$

This formula is used to obtain the matrices defined by $N\text{-mfr}_{ikp}$ and $N\text{prox}_{ikp}$.

Equation (14) is the multi-objective QAP formulation which is a minimization type function. Equations (15) and (16) are the assignment constraints, ensuring that each location contains only one cell and each cell is assigned to only one location in period p. The remaining constraints are the relationship of decision variables c_{ijp}, b_{ijlp}, and a_p. In the objective function "w" is the load factor of the qualitative part $Z2$, which is adopted from reference [10]. In his article, Urban suggested to select the value of w as the largest value of the material flow matrix $[\text{mfr}_{ikp}]_{N\times N(p)}$ in order to transform the qualitative term to correspond with the quantitative volumes. However, Urban did not utilize the normalization of the quantitative or qualitative term as it is done in other contemporary articles [11, 40]. Due to the adoption of that phenomenon, both the qualitative and quantitative objectives are transformed into the same scale. Thus, the value of the constant w is fixed in the range $0\leq w\leq 1$. If w takes the value 0, it means that the cost function transformed purely into material handling cost and if it takes value 1, then both material flow and relative proximity factor would share the same importance. However, in the experimental stage, the QAP model of cost function is tested for four values of w, which are 0.2, 0.4, 0.6, and 0.8.

3 Research methodologies

Genetic algorithm (GA) is an extensive, parallel, stochastic search, and optimisation technique, grounded on the perceptions of natural selection [41] and population genetics [42]. Holland [43] first proposed GA, and Goldberg [44] further made this algorithm accustomed among researchers. GA is implemented iteratively on a set of encoded chromosomes, called a population, with three basic genetic operators: selection, crossover, and mutation. Each chromosome is represented by a sequence, which could be binary or real coded. GA utilizes only the objective function information and probabilistic transition rules for genetic operations. A comprehensive theory of GA can be learned from the book compiled by Gen and Cheng [45].

SA is one of the oldest species among metaheuristics. The SA algorithm simulates the physical annealing process, where particles of a solid arrange themselves into a thermal equilibrium. An introduction to SA can be found in the book by Aarts and Korst [46]. The algorithm uses a predefined neighborhood structure of a set of feasible solutions. A control parameter

which is called "temperature" in analogy to the physical annealing process governs the search behavior. Each of the temperature pieces of the algorithm computes neighbor solutions to the current solution. If the new solution has a better objective function value than the old one, the new solution is "accepted"; else if the generated solution has a worse objective function value than the previous one, then it is only accepted with a certain probability depending on (i) the difference of the

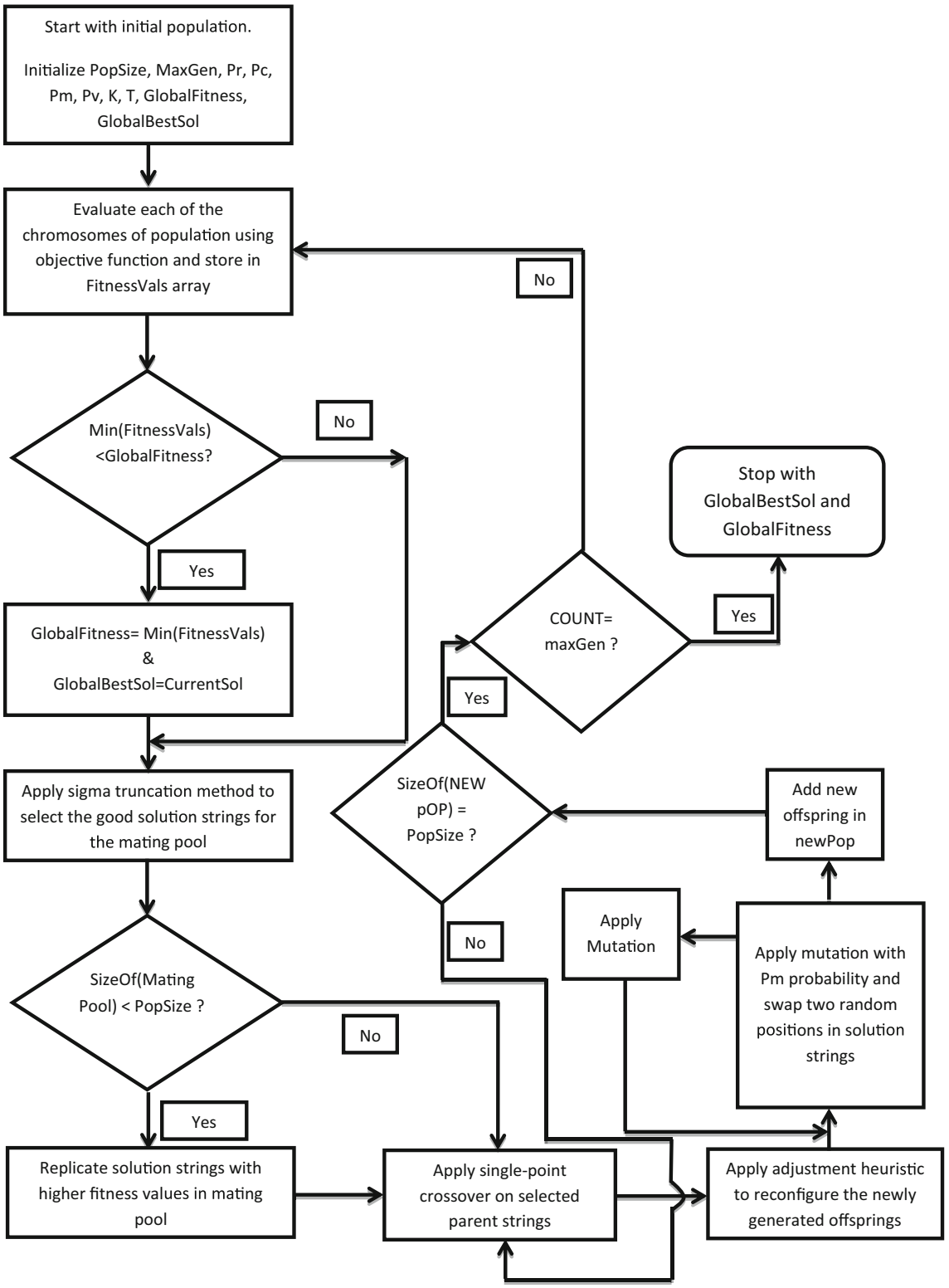

Fig. 4 IGA flowchart

objective function values in old and new solutions and (ii) the temperature parameter.

An elaborated discussion of the GA and SA methodologies developed for MPCLP is contributed accordingly in the following subsections.

3.1 Improved genetic algorithm

IGA starts with an initial population of randomly selected feasible solutions which are encoded on a finite length depending upon the number of cells in the layout. IGA is real coded and depends on the nature of the layout solution.

3.1.1 Initial population and encoding scheme

The initial population of solutions is generated randomly following some rules:

a) Every cell is placed in one location; thus, there would not be a repeat of string element. Therefore, encoding of a layout for a period can be presented as follows:

5	4	1	3	6	2

It is based on a 6 cells, 6 locations problem. It implies that cells 5, 4, 1, 3, 6, 2 are assigned to locations 1 2, 3, 4, 5, and 6, respectively.

b) Since the problem considered in this study is dynamic in nature, every period of the planning horizon would utilize different layout. Therefore, the initial solution string for a five periods problem would be as follows:

1						**2**						**3**						**4**						**5**					
5	4	1	3	6	2	2	4	6	1	3	5	1	5	4	2	6	3	3	6	4	1	5	2	4	5	1	3	6	2

It is also known as *chromosome*, which prominently shows five layout configurations for consecutive five periods. This arrangement is considered as a solution string of size 30 for a (6×6×5) problem. Once the initial population is created, each chromosome is then converted into a 6×5 array as follows:

5	4	1	3	6	2
2	4	6	1	3	5
1	5	4	2	6	3
3	6	4	1	5	2
4	5	1	3	6	2

3.1.2 Fitness function

The fitness function essentially evaluates a solution string by computing a numerical score. The MPCLP objective function (Eq. (15)) is used for this purpose. Since the MPCLP function is minimization type, therefore, the low score of a solution string is desirable in this context.

3.1.3 Selection method

Fitness-proportionate selection scheme frequently forces the algorithm to "exploit" the good areas at the cost of investigating the other parts of the solution space. At the later stage, when all the solution strings of the population are relatively similar (trivial variation in fitness), selection becomes insignificant, and the evolution stops with premature convergence [47]. To avoid such occurrence, the "sigma truncation" method is adopted as a selection method [44]. It eventually maintains a robust selection pressure throughout the direction of the execution. Unprocessed fitness values are first transformed into its expected value, which is a function of its fitness, the mean value, and standard deviation of population. The formula used in this work is as follows:

$$E(i,t) = \begin{cases} 1 + \dfrac{f(i) - f_{\mu}(t)}{C_{\sigma} \times f_{\sigma}(t)} & \text{if } f_{\sigma}(t) \neq 0 \\ 1 & \text{if } f_{\sigma}(t) \neq 0 \end{cases} \quad (24)$$

$E(i, t)$ is the expected value of individual i at time t, $f(i)$ is the fitness of i, $f_{\mu}(t)$ is the mean fitness of the population at time t, $f_{\sigma}(t)$ is the standard deviation of the population fitnesses at time t, and C_{σ} is the sigma truncating coefficient. A higher value of C_{σ} can reduce the fitness pressure of the population; thus, it is set to 3 throughout the experimental stage. This phenomenon would help the better solution to stand out more in the later stage of execution when the population is likely to converge and standard deviation is comparatively low. Therefore, the evolution continues.

3.1.4 Implementation of reproduction operator

Three genetic operations are employed in IGA implementation. These are reproduction, crossover, and

mutation. The reproduction operation triggers best fit chromosomes from the current population and put them into a list called "elite list" or "mating pool" to be used for next operations in the evolution. If the current population contains n chromosomes and the reproduction rate is defined by P_r, then $n \times P_r$ best chromosomes are required to be reproduced and to be put in elite list. Thereafter, $(n - n \times P_r)$ chromosomes are replicated from the list of "elite" chromosomes and added to the mating pool.

3.1.5 Implementation of crossover operator

The crossover operator exchanges genetic features between two parent chromosomes selected randomly from mating pool and then produces offsprings or child chromosomes. If the mating pool contains n chromosomes and the crossover rate is P_c, then $n \times P_c$ chromosomes randomly chosen for crossover. The crossover method applied in this IGA is based on *single-point* operation. An example of the crossover operation is depicted in Fig. 1.

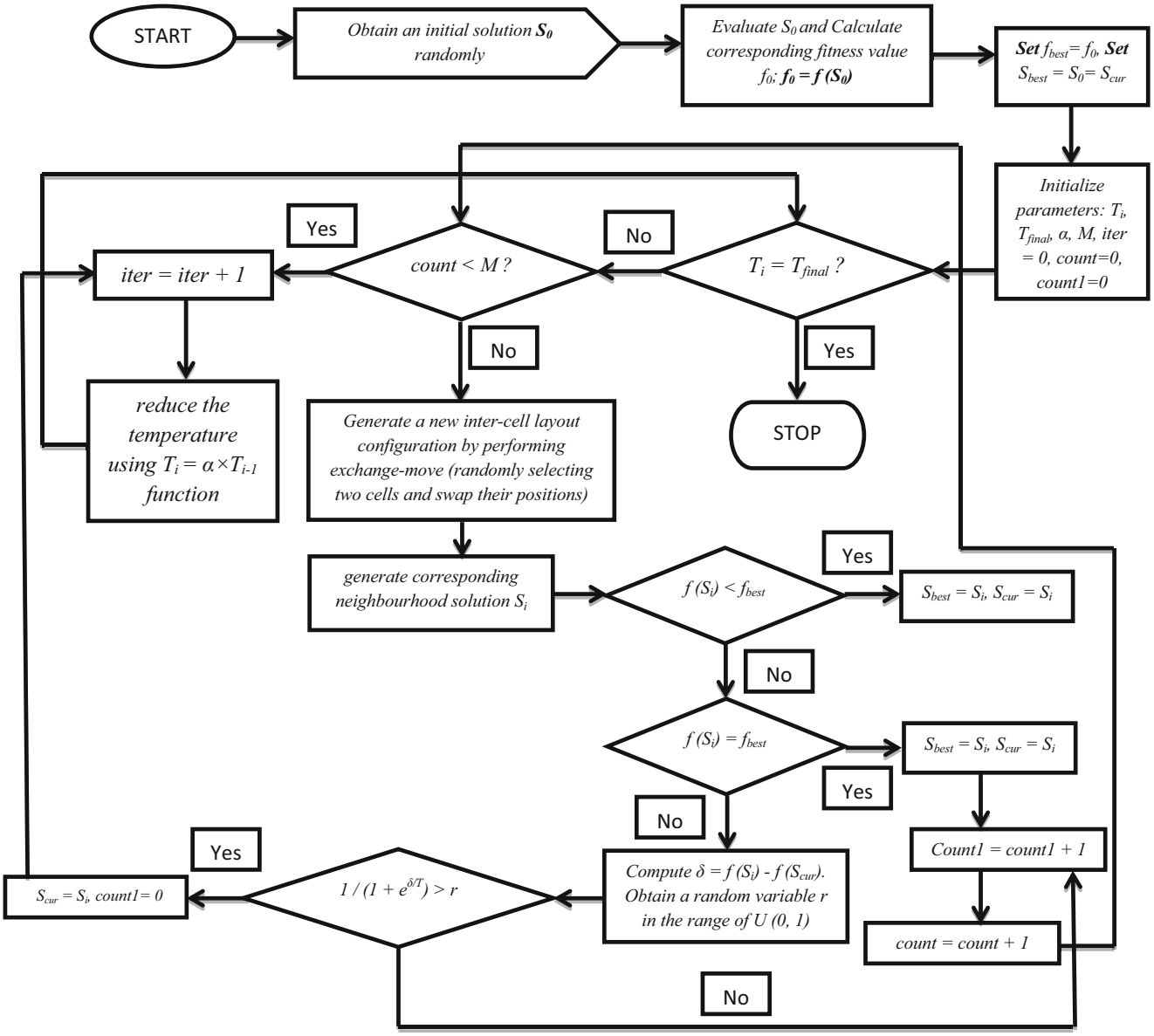

Fig. 5 SAH flowchart

Table 4 Parameter values for the IGA and SAH

Problem		IGA					SAH		
No. of cells	No. of periods	PopSize	MaxGen	P_r	P_c	P_m	T_0	α	M
6	2	30	500	0.6	0.65	0.01	10	0.95	300
	3	30	500	0.6	0.65	0.01	10	0.95	300
	5	30	500	0.6	0.65	0.01	10	0.9	250
	10	30	500	0.6	0.65	0.01	10	0.85	250
12	2	25	500	0.6	0.75	0.05	10	0.95	300
	3	25	500	0.6	0.75	0.05	10	0.95	300
	5	30	300	0.6	0.75	0.05	10	0.9	250
	10	30	300	0.6	0.75	0.05	10	0.85	250
24	2	20	350	0.6	0.8	0.1	10	0.95	250
	3	20	350	0.6	0.8	0.1	10	0.95	250
	5	25	300	0.6	0.8	0.1	10	0.9	200
	10	25	300	0.6	0.8	0.1	10	0.85	200

The crossover point is chosen randomly by generating a random integer number r between 0 and m (m is the number of cells in the layout) and exchanging the rth to mth segment of each of the parents. As shown in Fig. 1, $r=4$; thus, 4th to 6th genes are transferred between parents and two offsprings are generated.

At this moment, each of the offsprings will have five different layouts of five consecutive periods in which cell numbers are repeated which is not desirable. In order to avoid such situation, a small heuristic procedure is introduced to restructure the offsprings. The pseudocode is furnished as follows:

```
Step 1. Repeat
Step 2. For j=1 to m
Step 3. For k=j+1 to m
Step 4. If offspring1(j)==offspring1(k)
Step 5. Then do
Step 6. offspring1(j)=parent1(k)
Step 7. If offspring2(j)==offspring2(k)
Step 8. offspring2(j)=parent2(k)
Step 9. Until
Step 10. offspring1(j)!=offspring1(k)
```

Table 5 Results obtained for 6 cells×6 locations problems

Periods	Weight	IGA		SAH	
T	w	Fitness	CPU second	fitness	CPU second
2	0.2	*2.57315*	36.4384	2.60203	*17.8385*
	0.4	*3.33552*	34.1968	3.37097	*17.9187*
	0.6	*4.01722*	29.5701	4.13833	*17.5086*
	0.8	*4.72704*	27.3751	4.8222	*17.6332*
3	0.2	*2.64455*	41.1777	2.70485	*29.1721*
	0.4	*3.43554*	41.3803	3.50658	*29.0646*
	0.6	*4.20547*	40.5249	4.30211	*27.5990*
	0.8	*4.92037*	39.7005	5.04936	*27.4454*
5	0.2	*2.70240*	85.7489	2.71187	*75.2546*
	0.4	*3.47944*	86.0531	3.62772	*76.3102*
	0.6	*4.27922*	86.1153	4.38041	*77.1900*
	0.8	*4.96083*	85.2429	5.06534	*74.7217*
10	0.2	*2.83139*	308.1726	2.92312	*298.6552*
	0.4	*3.64459*	314.5878	3.76358	*310.1590*
	0.6	*4.43540*	311.5950	4.50004	*298.8098*
	0.8	*5.15573*	313.2379	5.30412	*298.0385*

Italicize values are the best obtained results

Table 6 Results obtained for 12 cells×12 locations problems

Periods	Weight	IGA		SAH	
T	w	Fitness	CPU second	Fitness	CPU second
2	0.2	*3.29436*	152.9942	3.38480	*110.5350*
	0.4	*4.26464*	152.3416	4.35158	*110.0463*
	0.6	*5.24337*	153.0419	5.28431	*111.1911*
	0.8	*6.10355*	154.3124	6.23516	*114.3423*
3	0.2	*3.36291*	*237.5744*	3.46637	245.4402
	0.4	*4.41385*	*247.4859*	4.43611	237.5707
	0.6	*5.35011*	*243.5326*	5.47250	234.9655
	0.8	*6.30749*	*238.8453*	6.43854	236.4380
5	0.2	*3.49779*	*564.6683*	3.54141	667.1761
	0.4	*4.48570*	*573.6178*	4.57992	635.4647
	0.6	*5.45233*	*559.2974*	5.58971	664.0384
	0.8	*6.46747*	*606.4231*	6.58335	637.5292
10	0.2	*3.59233*	*2070.3701*	3.67424	2666.4460
	0.4	*4.61552*	*2202.8450*	4.68706	2511.8531
	0.6	*5.65387*	*2186.5427*	5.71008	2500.9000
	0.8	*6.60422*	*2154.1022*	6.70108	2611.2724

Italicize values are the best obtained results

Table 7 Results obtained for 24 cells×24 locations problems

Periods	Weight	IGA		SAH	
T	*w*	Fitness	CPU second	Fitness	CPU second
2	0.2	*7.64312*	*705.0122*	7.66661	997.8511
	0.4	*9.78375*	*707.8895*	9.79841	991.5367
	0.6	*11.84873*	*704.6982*	11.8986	978.6104
	0.8	*13.97141*	*704.0253*	14.0063	972.3382
3	0.2	7.70680	*1083.7143*	*7.69732*	1253.9391
	0.4	*9.84152*	*1022.4629*	9.86685	1276.4802
	0.6	*12.01550*	*1087.2031*	12.03220	1278.9211
	0.8	*14.16888*	*1071.7117*	14.20060	1248.8362
5	0.2	*7.75302*	*3007.5651*	7.75465	4033.7234
	0.4	*9.90691*	*3023.1142*	9.92009	4082.8671
	0.6	*12.0838*	*3033.1395*	12.11380	4014.2551
	0.8	*14.2163*	*3082.5576*	14.25990	4022.4087
10	0.2	7.80185	*12,822.8631*	*7.80041*	19,319.2349
	0.4	*9.98855*	*12,790.4304*	10.01110	19,630.3872
	0.6	*12.13860*	*12,357.4678*	12.19823	19,692.7093
	0.8	*14.3554*	*13,072.5669*	14.40348	20,053.2231

Italicize values are the best obtained results

This procedure would eventually retain the exchanged genetic structure of the offsprings and facilitate in avoiding that undesirable situation. The resulting offsprings are portrayed in Fig. 2.

3.1.6 Implementation of mutation operator

The aim of mutation is to increase variability in chromosome of the population and to direct the population into an unexplored area of the search space. It often directs the execution to escape from local optima. If the population contains n chromosomes, m is the number of genes in each chromosome, and the rate of mutation is P_m, then $n \times m \times P_m$ chromosomes are randomly chosen for mutation. It generates new offsprings as demonstrated in Fig. 3.

It generates two random integers, $r1$ and $r2$ between 0 and m. Then swap the $r1$th and $r2$th genes as shown in Fig. 3. The steps of the mutation operation are presented as follows:

Step 1. *Generate* a random integer $r1$ lies in the range of *0 to m*
Step 2. *Generate* a random integer $r2$ lies in the range of 0 to m
Step 3. *Assign* $r1$th gene of the chromosome to a temporary variable *Temp*
Step 4. *Assign* $r2$th gene of the chromosome to the $r1$th gene's position
Step 5. *Assign* value assigned in *Temp* to the original position of $r2$th gene

3.1.7 Stopping condition

The stopping condition governs the execution of the IGA algorithm. It implies that all the operators are believed to be executed repeatedly until a stopping condition is encountered. The execution of IGA is eventually terminated once it reaches the maximum number of generations count. The flowchart of IGA is depicted in Fig. 4.

3.2 Simulated annealing heuristic

According to Kirkpatrick et al. [48], a SAH is a probabilistic local search heuristic algorithm. This technique, developed for discrete optimization, starts with an initial feasible solution to the problem and performs various walks in the search space according to the predefined annealing schedule.

Table 8 Fitness variations for 2 periods' problems by IGA and SAH

Problem		IGA			SAH		
No. of cells	*w*	Best fitness	Mean fitness	Worst fitness	Best fitness	Mean fitness	Worst fitness
6	0.2	2.57315	2.57315	2.57315	2.60203	2.60203	2.60203
	0.4	3.33552	3.33766	3.33980	3.37097	3.37994	3.38891
	0.6	4.01722	4.02924	4.04127	4.13833	4.15036	4.16239
	0.8	4.72704	4.74793	4.76883	4.82220	4.86910	4.91600
12	0.2	3.29436	3.29436	3.29436	3.38480	3.38480	3.38480
	0.4	4.26464	4.26464	4.26464	4.35158	4.35579	4.36000
	0.6	5.24337	5.24724	5.25112	5.28431	5.29801	5.31170
	0.8	6.10355	6.10702	6.11050	6.23516	6.25808	6.28100
24	0.2	7.64312	7.64378	7.64445	7.66661	7.67742	7.68823
	0.4	9.78375	9.79352	9.80330	9.79841	9.96075	10.12310
	0.6	11.84873	11.85995	11.87118	11.89860	12.05150	12.20450
	0.8	13.97141	13.98071	13.99000	14.00630	14.24720	14.48820

Table 9 Fitness variations for 3 periods' problems by IGA and SAH

Problem		IGA			SAH		
No. of cells	w	Best fitness	Mean fitness	Worst fitness	Best fitness	Mean fitness	Worst fitness
6	0.2	2.64455	2.64455	2.64455	2.70485	2.70485	2.70485
	0.4	3.43554	3.43622	3.43691	3.50658	3.50674	3.50690
	0.6	4.20547	4.20945	4.21344	4.30211	4.34071	4.37932
	0.8	4.92037	4.92163	4.92290	5.04936	5.08968	5.13000
12	0.2	3.36291	3.36291	3.36291	3.46637	3.46637	3.46637
	0.4	4.41385	4.41602	4.41820	4.43611	4.47750	4.51890
	0.6	5.35011	5.35986	5.36962	5.47250	5.49175	5.51100
	0.8	6.30749	6.32424	6.34100	6.43854	6.46362	6.48870
24	0.2	7.70680	7.70690	7.70700	7.69732	7.70461	7.71190
	0.4	9.84152	9.86018	9.87885	9.86685	9.88133	9.89582
	0.6	12.01550	12.22360	12.4317	12.0322	12.17175	12.31131
	0.8	14.16888	14.27199	14.3751	14.2006	14.24765	14.29470

3.2.1 Initial feasible solution

At the very first step, an initial solution is generated randomly in the feasible region of the solution space which is the first point to begin with the search. In this article, the SAH algorithm is used to optimize the MPCLP problem. Its initial solution generates randomly by assigning each manufacturing cell to some location in every period by following the assignment constraints.

3.2.2 Generating a neighborhood solution

A neighborhood solution is generated by a swap operation performed on the solution string. First SAH generates two random integers, $r1$ and $r2$ between 0 and m. Then swap the $r1$th and $r2$th elements as demonstrated in the mutation operation of IGA (Section 3.1.6).

3.2.3 Initial temperature and annealing schedule

An initial temperature T_0 and an annealing (cooling) schedule α are required to influence the sequence of walks in the search space. Ordinarily, the initial temperature is prefixed to some higher value which accepts all the solutions virtually. Cooling schedule is defined by α, which is the rate at which the temperature is reduced to the freezing temperature in order to minimize probability of acceptance of inferior quality solutions. Souilah reported several cooling functions which could be used as temperature reduction functions [49]. In present research, temperature is reduced following a geometric function:

$$T = \alpha \times T_0 \tag{25}$$

where α varies with the size of the problem considered.

Table 10 Fitness variations for 5 periods' problems by IGA and SAH

Problem		IGA			SAH		
No. of cells	w	Best fitness	Mean fitness	Worst fitness	Best fitness	Mean fitness	Worst fitness
6	0.2	2.70240	2.70240	2.70240	2.71187	2.71187	2.71187
	0.4	3.47944	3.47972	3.48000	3.62772	3.62791	3.62810
	0.6	4.27922	4.28537	4.29153	4.38041	4.38676	4.39312
	0.8	4.96083	5.02855	5.11039	5.06534	5.15456	5.22475
12	0.2	3.49779	3.49779	3.49779	3.54141	3.54141	3.54141
	0.4	4.48570	4.49965	4.51360	4.57992	4.58812	4.59632
	0.6	5.45233	5.46966	5.48700	5.58971	5.61090	5.63209
	0.8	6.46747	6.48051	6.49354	6.58335	6.63674	6.69013
24	0.2	7.75302	7.75302	7.75302	7.75465	7.75676	7.75887
	0.4	9.90691	9.92035	9.93380	9.92009	9.92472	9.92936
	0.6	12.08380	12.12096	12.15812	12.11380	12.15217	12.19054
	0.8	14.21630	14.24275	14.26920	14.25990	14.28845	14.31700

Table 11 Fitness variations for 10 periods' problems by IGA and SAH

Problem		IGA			SAH		
No. of cells	w	Best fitness	Mean fitness	Worst fitness	Best fitness	Mean fitness	Worst fitness
6	0.2	2.83139	2.83139	2.83139	2.92312	2.92312	2.92312
	0.4	3.64459	3.64459	3.64459	3.76358	3.76399	3.76441
	0.6	4.43540	4.43875	4.44210	4.50004	4.50100	4.50196
	0.8	5.15573	5.16346	5.17119	5.30412	5.33311	5.36210
12	0.2	3.59233	3.59672	3.60112	3.67424	3.67456	3.67488
	0.4	4.61552	4.61717	4.61883	4.68706	4.68853	4.69000
	0.6	5.65387	5.65743	5.66100	5.71008	5.75111	5.79213
	0.8	6.60422	6.61279	6.62137	6.70108	6.90656	7.11204
24	0.2	7.80185	7.80188	7.80191	7.80041	7.80084	7.80127
	0.4	9.98855	9.98893	9.98932	10.0111	10.09606	10.18103
	0.6	12.13860	12.14138	12.14416	12.19823	12.20136	12.20450
	0.8	14.35540	14.35741	14.35942	14.40348	14.44584	14.48820

3.2.4 Length of Markov Chain

Markov chain length (M) signifies the maximum number of tries over which the annealing procedure attains the thermal equilibrium state for a piece of temperature. The value of M is set between 200 and 300 after performing a set of preliminary tests, depending upon the problem size.

3.2.5 Acceptance probability

To decide the tolerance level of accepting a worse solution, in a temperature piece T, this acceptance probability plays an important role. If f is the value of current energy state of the annealing process, a new layout arrangement or "neighbor" is selected at random, which gives a new energy level f_{new}. If $f_{new} \leq f$, it accepts the new arrangement and f_{new} becomes the current energy level and new arrangement becomes the current best solution. Otherwise, the assignment is only accepted randomly with probability:

$$p = \frac{1}{\left[1 + e^{\frac{(f - f_{new})}{k \times T}}\right]} \tag{26}$$

It is also known as *modified Boltzmann's acceptance probability* (k=Boltzmann's constant). Thus, if f_{new} is close to f, the arrangement is more likely to be accepted. If the temperature is high, the exponent will be close to zero, and so the probability will be close to 1. As the temperature approaches zero, the exponent approaches -∞, and the probability approaches zero.

3.2.6 Stopping condition

The SA algorithm can be terminated, when the current temperature reaches to freezing point (very small temperature which is close to zero) or if the neighbor solution is not being improved after a long period.

The flowchart of SA algorithm is presented in Fig. 5.

4 Computational results

This section contains a structured discussion based on the performance of each of the proposed methodologies, IGA and SAH. The aim is also to portray a comparative study between IGA and SAH in terms of solution quality and computational time. In order to verify the QAP model of MPCLP and state the effectiveness of both the proposed algorithms, test problems are required. Even though there are many articles available in the

Table 12 Experimental setup of statistical testing of IGA and SAH

IGA	SAH	Difference (d)
4.96083	5.06534	−0.10488
4.98919	5.08551	−0.09632
4.98941	5.09112	−0.10171
4.99933	5.10922	−0.10989
5.00843	5.11382	−0.10539
5.01143	5.1248	−0.11337
5.01959	5.1288	−0.10921
5.02143	5.13516	−0.11373
5.025	5.14423	−0.11923
5.02712	5.14956	−0.12244
5.02879	5.15648	−0.12769
5.02961	5.1803	−0.15069
5.03247	5.18382	−0.15135
5.04306	5.18598	−0.14292
5.04543	5.18921	−0.14378
5.04853	5.19357	−0.14504
5.05533	5.19587	−0.14054
5.0649	5.20843	−0.14353
5.07424	5.21308	−0.13884
5.08602	5.22475	−0.13873
	Mean difference (μ_d)	−0.125964
	Standard deviation (σ^2)	0.018234383

Table 13 Normality test results of data for IGA and SAH (Anderson-Darling method)

Parameters	Values for IGA	Values for SAH
A-squared	0.146	0.325
p value	0.960	0.502
95 % critical value	0.787	0.787
99 % critical value	1.092	1.092

Table 14 Two-sample F-test for variances of IGA and SAH (α =0.05)

	IGA	SAH
Mean	*5.0279885*	5.153953
Variance	*0.000934133*	0.002143
Observations	20	20
df	19	19
F	0.44	
$P(F \leq f)$	One-tail=0.039	Two-tail=0.078
$F_{critical}$	One-tail=2.17	Two-tail=2.53

Italicize values are the best obtained results

area of dynamic facility layout problems with small to large datasets, those data would not fit in the proposed MPCLP model of us. That is because we considered a new phenomenon called "relative proximity factor." Therefore, test problems are supposed to be simulated logically. To cater the requirement, datasets are generated in the range of 6×6×2 to 24×24×10, i.e., 6 cells with 2, 3, 5, and 10 periods problems, 12 cells with 2, 3, 5, and 10 periods problems, and 24 cells with 2, 3, 5, and 10 periods problems, specifically, 12 test problems. Material flow cost and cell reconfiguration costs are generated by means of uniform distribution following Balakrishnan and Cheng's procedure [50]. The cell-to-cell material flow is modified relatively to restrict the aggregate flow within a predefined level throughout the planning horizon for any problem. This approach further restricts the domination of one period over another. Cell reconfiguration costs were also tuned in such a way that the mean reconfiguration cost is not more than 10 % of the mean material flow cost. One example dataset of size 6×6×5 is shown in the Appendices 1 and 2. Proposed IGA and SAH algorithms are coded in Matlab 7.6 using an Intel PC with Quad CPU (2.83 GHz) and 4 GB of RAM. For all the test problems, computational complexities of the algorithms intensify exponentially with the number of planning periods. Therefore, a good design is indeed important while dealing with large test problems. The evaluation criteria of IGA and SAH are based on total material flow cost, total reconfiguration cost, and total numerical value of

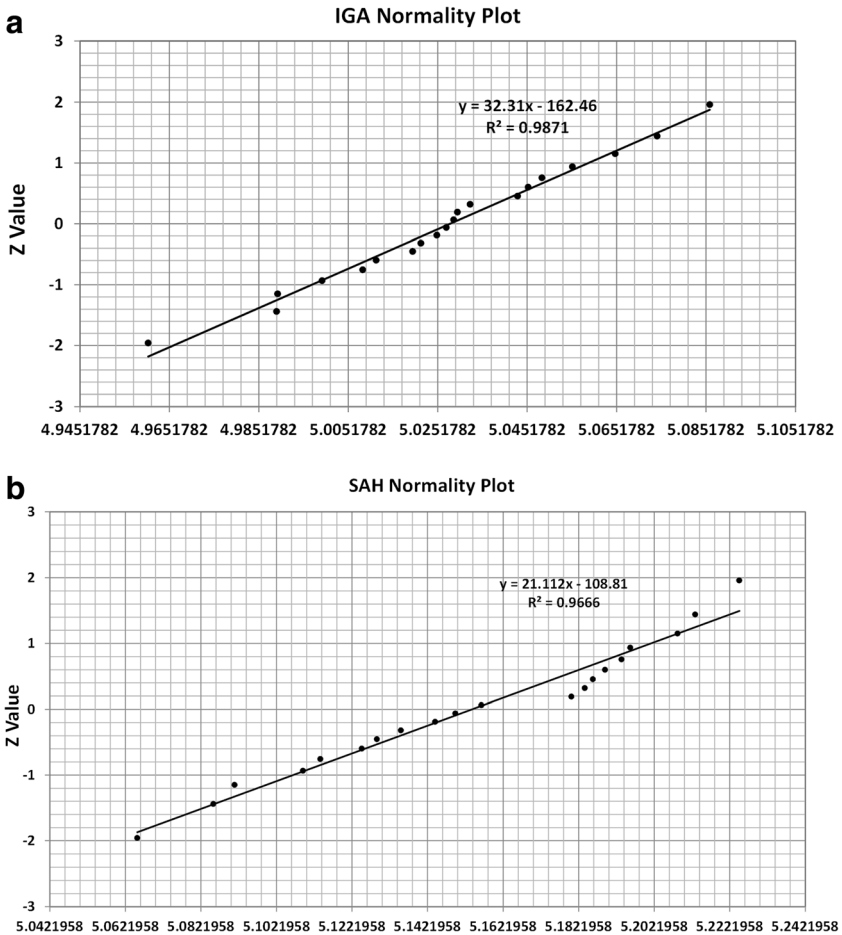

Fig. 6 **a** Normal probability plot for IGA. **b** Normal probability plot for SAH

Table 15 Two-sample *t*-test assuming equal variance and equal sample size of IGA and SAH (α=0.05)

	IGA	SAH
Mean	5.027989	5.153953
Variance	0.000934	0.002143
Observations	20	20
Pooled variance	0.001538	
Hypothesized mean difference	0	
df	38	
t-statistic	−10.156	
$P(T \le t)$ (one-tail)	0.000	
$T_{critical}$ (one-tail)	1.686	
$P(T \le t)$ (two-tail)	0.000	
$T_{critical}$ (two-tail)	2.024	

relative proximity factor. To determine the desired values of the parameters of IGA and SAH, extensive experiments are done with the test problems (at least 20 executions for each of the test instances). Parameters of IGA are the population size (*PopSize*), number of maximum generations (*MaxGen*), probability of reproduction (P_r), probability of crossover (P_c), and probability of mutation (P_m) and parameters of SAH are the initial temperature (T_0), cooling rate (α), and length of Markov chain (M), which are to be fixed to obtain good results. Details of the values of the parameters are reported in Table 4. For the mid to large size problems, the value of the parameters *PopSize*, *MaxGen*, α, and M are fixed moderately low in order to minimize the computational efforts and it is done after extensive analysis with different settings of parameters on all the datasets. The decision of keeping low CPU time is a crucial trade-off in the context of the QAP since the metaheuristic algorithms have a tendency to get trapped in local optima with reduced values of parameters.

Tables 5, 6, and 7 demonstrate the results obtained by proposed IGA and SAH techniques. Both the methodologies are compared in terms of objective values achieved and computational time utilized. Since the objective of this research is to attain near-optimal solutions within reasonable computational efforts, both of these issues are given equal importance to show the efficiency of the algorithms. Due to the inadequacy of such mathematical model in CMS literature, we were unable to assess our results with any of the past results obtained so far, which is the downside of our research. In order to fill the gap, we performed various analyses based on the performance of proposed IGA and SAH. A total of 48 test instances are executed for all the 12 test problems with four different values of load factor *w*. Out of these, IGA achieved the best results for 46 test instances in terms of solution quality and 28 test instances in terms of CPU time which further states a 95.83 and 58.33 % improvements, respectively. For the small size problems (6×6×2, 6×6×3, 6×6×2, and 12×12×2), SAH appeared to be quite efficient in terms of CPU time; for the mid to large size problems (12×12×3 to 12×12×10 and 24×24×2 to 24×24×10), the IGA technique outperforms SAH in terms of CPU time. Conversely, SAH is capable of obtaining solutions which are very close to the solutions obtained by IGA in terms of quality. The value of load factor *w* further determines the value of the total cost of the solutions. The objective value increases proportionally with the value of *w*.

Due to the probabilistic nature, these two techniques attain different solutions in a complete execution. Thus, we run each of the algorithms 20 times for every test problem for every value of *w* and the best out of those is selected as global best solution. Deviations in fitness values for all the test problems are depicted in Tables 8, 9, 10, and 11.

To prove the competence of IGA over SAH, an elaborated statistical analysis is performed on a sample test problem of size 6×6×5 with a prefixed value of w=0.8. The experimental setup is given in Table 12 where each of the two techniques is executed for 20 times and the objective values (data) obtained are recorded. First, we tested the normality of data using Anderson-Darling

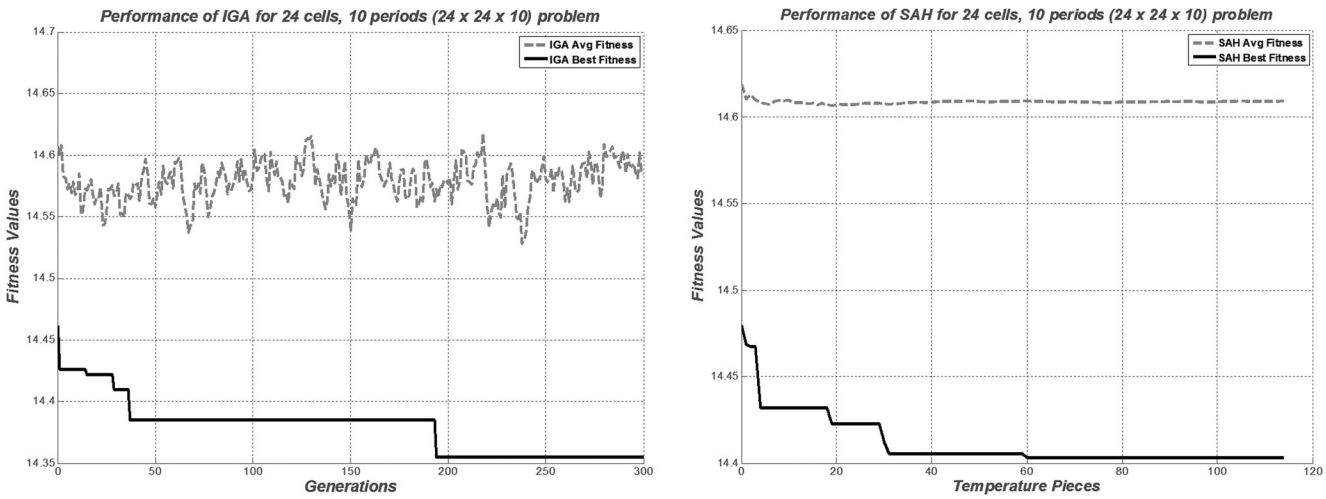

Fig. 7 Convergence analysis of IGA and SAH for 24 cells, 10 periods problem

normality test. The result is shown in Table 13. The null hypothesis used is *H*0, if the data is normal. It implies the rejection of the null hypothesis when *p* value $\leq\alpha$ or *A-squared*>critical value. We accept the null hypothesis due to the following:

- Dots fit the trend line on the normal probability plot (Fig. 6a, b).
- For IGA, *p* value=0.960>0.05 and A-squared=0.146< 0.787, and <1.092; therefore, we accept the null hypothesis.
- For SAH, *p* value=0.502>0.05 and A-squared=0.325< 0.787 and <1.092; therefore, we accept the null hypothesis.

Therefore, we could be at least 95 % confident that the data are normal.

Since the two sets of objective values are not *paired* or *dependent* in any case, we need to verify whether the variances are equal or not. Thus, *F*-test is performed and the results are depicted in Table 14.

The experimental result of the *F*-test can be illustrated as follows:

If the test statistic<critical value ($F<F_{\text{critical}}$), accept the null hypothesis; in other words, if *p* value> α, accept the null hypothesis. Table 13 depicts that $F<F_{\text{critical}}$ (0.44<2.17) and *p* value (*two-tail*)> α (0.078>0.05); thus, we accept the null hypothesis that the variances are equal. Now we conduct the *t*-test assuming equal variances. The result of *t*-test is reported in Table 15.

We can interpret the two-sample *t*-test results as follows:

If the test statistic<critical value ($t<t_{\text{critical}}$), accept the null hypothesis or in other words, if *p* value> α, accept the null hypothesis. Since the null hypothesis is that the mean difference=0, this is a two-sided test. Therefore, we use the two-tail values for the analysis. Since the *t*-statistic< t_{critical} (−10.156<1.686) and (−10.156<2.024) and both the *p* values< α (0<0.05), we reject the null hypothesis that the means are not the same. Therefore, we found data inconsistency in the experiments which further states the performance differences of two techniques, IGA and SAH. More specifically, it can be stated that IGA (mean= 5.027989) is an improved method than SAH (mean= 5.153953) while obtaining solution for the test problem of size 6×6×5 at a 95 % confidence level. If we perform this statistical analysis on all the test problems, similar outcomes can be found.

Both the proposed methods are extremely improved when compared with the latest published methodologies. Computational experiments of such comparison are illustrated in Appendix 2.

4.1 Convergence analysis of IGA and SAH

Figure 7 shows the example of convergence analysis of both the algorithms, which are almost equivalent for all the problem datasets. All the test problems with 10 planning periods are selected to demonstrate the convergence curve during iterations of the proposed IGA and SAH techniques. The IGA obtains best layout configurations which are better than SAH solutions. The average fitness values achieved by IGA are also improved in terms of quality. Moreover, the fitness curves demonstrate the competence of IGA to escape from local optimal solutions since it constantly improves its solution until the end of its generation counts. The proposed algorithms contribute the same pattern of convergence for all the tested problems; therefore, the convergence property is established.

5 Conclusions

We have presented a state-of-the-art mathematical programming model of inter-cell layout design in the area of CMS. We have adopted two criteria in the said model: (a) an adjacency-based relative proximity factor, which is essential in determining the combined effect of closeness and separation factors between two cells and (b) the dynamic behavior of the cell layout design, which is practical when product mix is considered and cell reconfiguration cost is taken into consideration. The multi-objective dynamic layout problem has been recently developed for generalized facility layout problem [51, 52]. However, it has never been practised in the history of CMS research to the best of our knowledge. Therefore, the test datasets are not available in the past literature for the said problems. To cater that purpose, 12 test problems of size 6× 6×2 to 24×24×10 are simulated using the method stated in [50]. Due to the NP-hard nature of these problems, two latest metaheuristic algorithms, namely, IGA and SAH, are developed in order to obtain the near-optimal solutions. Results obtained by both the techniques are viable in terms of solution quality and CPU time. Figure 5 shows that IGA is capable of obtaining solutions, consuming lesser CPU time exclusively for mid to large size problems. Out of 48 solution instances, IGA obtained 95.83 % improved results. The test results are validated using Anderson-Darling's normality test to depict that the solution data are normally distributed (Fig. 7a, b). Further, the statistical tests (*f*-test and *t*-test) are carried out to prove the competence of the proposed IGA over SAH. Therefore, we authorize the feasibility of the stated mathematical model and proposed algorithms which can be efficiently used in the realistic production situations to obtain efficient layouts over multiple planning horizons. The possible extension of this research would be to incorporate intra-cell layout concept by considering part demands, batch sizes, machining sequences, and machine similarities in the stated model formulation to develop a more generalized and realistic QAP paradigm for a complete layout design problem in CMS.

Acknowledgments The corresponding author Tamal Ghosh (IF120670) is thankful to DST-INSPIRE Division, Department of Science & Technology, Government of India for the financial support to carry out the Doctoral Research.

Appendix 1

An example of the normalized matrices of the 6×6×5 test problem and its solution using IGA is depicted hereunder. The value of load factor *w* is considered as 0.8.

Material flow cost:

Period 1:

	C1	*C2*	*C3*	*C4*	*C5*	*C6*
C1	0	0.011494	0.011494	0.011494	0.017241	0.011494
C2	0.011494	0	0	0.005747	0.005747	0.005747
C3	0.011494	0	0	0.005747	0.011494	0.005747
C4	0.011494	0.005747	0.005747	0	0.011494	0.005747
C5	0.017241	0.005747	0.011494	0.011494	0	0.005747
C6	0.011494	0.005747	0.005747	0.005747	0.005747	0

Period 2:

	C1	*C2*	*C3*	*C4*	*C5*	*C6*
C1	0	0	0	0	0.005747	0
C2	0	0	0	0	0.005747	0
C3	0	0	0	0.011494	0.005747	0
C4	0	0	0.011494	0	0.005747	0
C5	0.005747	0.005747	0.005747	0.005747	0	0.005747
C6	0	0	0	0	0.005747	0

Period 3:

	C1	*C2*	*C3*	*C4*	*C5*	*C6*
C1	0	0.005747	0.005747	0.005747	0	0.011494
C2	0.005747	0	0.005747	0.011494	0.005747	0.011494
C3	0.005747	0.005747	0	0.011494	0	0.005747
C4	0.005747	0.011494	0.011494	0	0.005747	0.017241
C5	0	0.005747	0	0.005747	0	0.011494
C6	0.011494	0.011494	0.005747	0.017241	0.011494	0

Period 4:

	C1	*C2*	*C3*	*C4*	*C5*	*C6*
C1	0	0	0.005747	0	0.011494	0.005747
C2	0	0	0.005747	0.011494	0.017241	0.005747
C3	0.005747	0.005747	0	0.005747	0.011494	0
C4	0	0.011494	0.005747	0	0.011494	0
C5	0.011494	0.017241	0.011494	0.011494	0	0.011494
C6	0.005747	0.005747	0	0	0.011494	0

Period 5:

	C1	*C2*	*C3*	*C4*	*C5*	*C6*
C1	0	0.017241	0.011494	0.011494	0.011494	0.005747
C2	0.017241	0	0.005747	0.005747	0.005747	0.005747
C3	0.011494	0.005747	0	0.005747	0.005747	0
C4	0.011494	0.005747	0.005747	0	0.011494	0.005747
C5	0.011494	0.005747	0.005747	0.011494	0	0.005747
C6	0.005747	0.005747	0	0.005747	0.005747	0

Relative proximity factor: (considered identical for all the periods)

Period 1:

	C1	*C2*	*C3*	*C4*	*C5*	*C6*
C1	0	0.006667	0	0.01	0.01	0.003333
C2	0.006667	0	0	0.003333	0.013333	0.01
C3	0	0	0	0.006667	0.006667	0.006667
C4	0.01	0.003333	0.006667	0	0.006667	0.003333
C5	0.01	0.013333	0.006667	0.006667	0	0.013333
C6	0.003333	0.01	0.006667	0.003333	0.013333	0

Period 2:

	C1	*C2*	*C3*	*C4*	*C5*	*C6*
C1	0	0.006667	0	0.01	0.01	0.003333
C2	0.006667	0	0	0.003333	0.013333	0.01
C3	0	0	0	0.006667	0.006667	0.006667
C4	0.01	0.003333	0.006667	0	0.006667	0.003333
C5	0.01	0.013333	0.006667	0.006667	0	0.013333
C6	0.003333	0.01	0.006667	0.003333	0.013333	0

Period 3:

	C1	*C2*	*C3*	*C4*	*C5*	*C6*
C1	0	0.006667	0	0.01	0.01	0.003333
C2	0.006667	0	0	0.003333	0.013333	0.01
C3	0	0	0	0.006667	0.006667	0.006667
C4	0.01	0.003333	0.006667	0	0.006667	0.003333
C5	0.01	0.013333	0.006667	0.006667	0	0.013333
C6	0.003333	0.01	0.006667	0.003333	0.013333	0

Period 4:

	C1	*C2*	*C3*	*C4*	*C5*	*C6*
C1	0	0.006667	0	0.01	0.01	0.003333
C2	0.006667	0	0	0.003333	0.013333	0.01
C3	0	0	0	0.006667	0.006667	0.006667
C4	0.01	0.003333	0.006667	0	0.006667	0.003333
C5	0.01	0.013333	0.006667	0.006667	0	0.013333
C6	0.003333	0.01	0.006667	0.003333	0.013333	0

Period 5:

	C1	*C2*	*C3*	*C4*	*C5*	*C6*
C1	0	0.006667	0	0.01	0.01	0.003333
C2	0.006667	0	0	0.003333	0.013333	0.01
C3	0	0	0	0.006667	0.006667	0.006667
C4	0.01	0.003333	0.006667	0	0.006667	0.003333
C5	0.01	0.013333	0.006667	0.006667	0	0.013333
C6	0.003333	0.01	0.006667	0.003333	0.013333	0

Cell-to-location assignment cost:

Period 1:

	L1	*L2*	*L3*	*L4*	*L5*	*L6*
C1	0.002217	0.008869	0	0.004435	0.003326	0.005543
C2	0.008869	0.008869	0.009978	0.006652	0.003326	0.011086
C3	0.003326	0.009978	0.004435	0.001109	0.005543	0.008869
C4	0.004435	0.001109	0	0.002217	0.003326	0.011086
C5	0.001109	0.005543	0.008869	0.001109	0.009978	0.002217
C6	0.009978	0.006652	0.001109	0.003326	0.008869	0.005543

Period 2:

	L1	*L2*	*L3*	*L4*	*L5*	*L6*
C1	0.009978	0	0.004435	0	0.006652	0.004435
C2	0.011086	0.011086	0.004435	0.005543	0.006652	0.011086
C3	0	0.003326	0.001109	0.007761	0.001109	0
C4	0.004435	0.006652	0.004435	0.002217	0	0.003326
C5	0.006652	0.011086	0.001109	0.009978	0.002217	0.005543
C6	0.007761	0.002217	0.006652	0.011086	0.009978	0

Period 3:

	L1	*L2*	*L3*	*L4*	*L5*	*L6*
C1	0.004435	0.004435	0.003326	0.002217	0.006652	0
C2	0	0	0.009978	0.004435	0	0.001109
C3	0.002217	0.009978	0.004435	0.001109	0.002217	0.007761
C4	0.001109	0.011086	0.001109	0.001109	0.003326	0.008869
C5	0.002217	0.005543	0.008869	0.011086	0.009978	0.007761
C6	0.002217	0.005543	0.004435	0.011086	0	0.004435

Period 4:

	L1	*L2*	*L3*	*L4*	*L5*	*L6*
C1	0.008869	0.006652	0.007761	0.005543	0.001109	0.011086
C2	0.007761	0.005543	0.006652	0	0.009978	0.011086
C3	0.006652	0.007761	0	0.002217	0.007761	0.011086
C4	0.007761	0.002217	0.006652	0.006652	0.004435	0.008869
C5	0.004435	0.009978	0.003326	0.008869	0.007761	0.007761
C6	0.003326	0.009978	0	0.011086	0.002217	0.006652

Period 5:

	L1	*L2*	*L3*	*L4*	*L5*	*L6*
C1	0.008869	0.006652	0.007761	0.005543	0.001109	0.011086
C2	0.007761	0.005543	0.006652	0	0.009978	0.011086
C3	0.006652	0.007761	0	0.002217	0.007761	0.011086
C4	0.007761	0.002217	0.006652	0.006652	0.004435	0.008869
C5	0.004435	0.009978	0.003326	0.008869	0.007761	0.007761
C6	0.003326	0.009978	0	0.011086	0.002217	0.006652

Fixed reconfiguration cost:

P1	*P2*	*P3*	*P4*	*P5*
0	0.25	0.25	0.25	0.25

Variable reconfiguration cost:

	P1	*P2*	*P3*	*P4*	*P5*
C1	0	0.039894	0.034574	0.050532	0.055851
C2	0	0.029255	0.047872	0.031915	0.042553
C3	0	0.055851	0.045213	0.034574	0.031915
C4	0	0.026596	0.050532	0.037234	0.042553
C5	0	0.034574	0.055851	0.047872	0.031915
C6	0	0.037234	0.042553	0.039894	0.053191

Near-optimal solutions obtained:

Period	*Physical layout arrangements*	*Optimal cost*
1	**6 3 2** **5 1 4**	**4.96083**
2	**2 1 6** **5 3 4**	
3	**1 3 5** **4 6 2**	
4	**3 6 4** **5 1 2**	
5	**1 3 4** **5 2 6**	

Appendix 2

In order to prove the competence of the proposed IGA and SAH over the latest published metaheuristic techniques such as other GA or SA variants, a computational experiment is performed over few datasets and it is shown that both the methods proposed in this article are extremely proficient and outperformed the latest SA [53] and GA [54] techniques. The results are demonstrated in Table 16.

Table 16 Comparison of Immune-GA-RS with other published techniques

Number of cells	SA (S&S-2008) [53]	SAH	GA-(KTFD-2011) [54]	IGA
Nug5	*25*	*25*	*25*	*25*
Nug6	*43*	*43*	*43*	*43*
Nug7	*74*	*74*	*74*	*74*
Nug8	*107*	*107*	*107*	*107*
Nug12	302	*289*	300	*289*
Nug15	640	*575*	601	*575*
Nug20	1474	1287	1310	*1285*
Nug30	3518	3097	3128	*3062*

Italicize values are the best obtained results

All the test problems (Nug5–Nug30) considered for this experiment are available in *QAPLIB-A Quadratic Assignment Problem Library* (http://www.seas.upenn.edu/qaplib/). The global optimal solutions are also available in that stated library. Table 16 clearly states that IGA outperforms latest GA proposed in [54] and attains 50 % better results, while SAH is also extremely capable of achieving 25 % better results which are optimal according to the QAPLIB. Thus, it is proved that the proposed methods are improved and better.

References

1. Burbidge JL (1977) A manual method of production flow analysis. Production Engineer 56:34–38
2. Selim MS, Askin RG, Vakharia AJ (1998) Cell formation in group technology: review evaluation and directions for future research. Comput Ind Eng 34:3–20
3. Won Y, Currie KR (2007) Fuzzy ART/RRR-RSS: a two-phase neural network algorithm for part-machine grouping in cellular manufacturing. Int J Prod Res 45:2073–2104
4. Heragu SS, Kakuturi SR (1997) Grouping and placement of machine cells. IIE Trans 29:561–571
5. Kulkarni PC, Shanker K (2007) A genetic algorithm for layout problems in cellular manufacturing systems. Proceedings of the 2007 I.E. IEEM pp 694–698
6. Balakrishnan J, Cheng CH (2009) The dynamic plant layout problem: incorporating rolling horizons and forecast uncertainty. Omega 37:165–177
7. Wang S, Sarker BR (2002) Locating cells with bottleneck machines in cellular manufacturing systems. Int J Prod Res 40:403–424
8. Kaufman L, Broeckx F (1978) An algorithm for the quadratic assignment problem using benders' decomposition. Eur J Oper Res 2: 207–211
9. Fortenberry JC, Cox JF (1985) Multiple criteria approach to the facility layout problem. Int J Prod Res 23:773–782
10. Urban TL (1987) A multiple criteria model for the facility layout problem. Int J Prod Res 25:1805–1812
11. Harmonosky CM, Tothero GK (1992) A multi-factor plant layout methodology. Int J Prod Res 30:1773–1789
12. Nicol LM, Hollier RH (1983) Plant layout in practice. Material Flow 1:177–188
13. Rosenblatt JM (1986) The dynamics of plant layouts. Manag Sci 32:76–86
14. Pillai MV, Hunagund IB, Krishnan KK (2011) Design of robust layout for dynamic plant layout problems. Comput Ind Eng 61: 813–823
15. Ripon KSN, Glette K, Koch D, Hovin M, Torresen J (2012) Genetic algorithm using a modified backward pass heuristic for the dynamic facility layout problem. PALADYN. doi:10.2478/s13230-012-0008-1
16. Lacksonen TA, Enscore EE (1993) Quadratic assignment algorithms for the dynamic layout problems. Int J Prod Res 31:503–517
17. Kaku KB, Mazzola JB (1997) A tabu search heuristic for the dynamic plant layout problem. INFORMS J Comput 9:374–384
18. Urban TL (1998) Solution procedures for the dynamic facility layout problem. Ann Oper Res 76:323–342
19. Sahin R, Ertogral K, Türkbey O (2010) A simulated annealing heuristic for the dynamic layout problem with budget constraint. Comput Ind Eng 59:308–313
20. Ripon KSN, Glette K, Hovin M, Torresen J (2010) An adaptive local search based genetic algorithm for solving multi- objective facility layout problem. Neural information processing: theory and algorithms, lecture notes in computer science, vol 6443, pp 540–550
21. Logendran R (1991) Impact of sequence of operations and layout of cells in cellular manufacturing. Int J Prod Res 29:375–390
22. Alfa AS, Chen M, Heragu SS (1992) Integrating the grouping and layout problems in cellular manufacturing systems. Comput Ind Eng 23:55–58
23. Sarker BR, Yu JA (1994) Two-phase procedure for duplicating bottleneck machines in a linear layout, cellular manufacturing system. Int J Prod Res 32:2049–2067
24. Tang C, Abdel-Malek LL (1996) A framework for hierarchical interactive generation of cellular layout. Int J Prod Res 34:2133–2163
25. Lee SD (1998) Configuring layout for a cellular manufacturing system. Int J Syst Sci 29:557–564
26. Salum L (2000) The cellular manufacturing layout problem. Int J Prod Res 38:1053–1069
27. Chan WM, Chan CY, Kwong CK (2004) Development of the MAIN algorithm for a cellular manufacturing machine layout. Int J Prod Res 42:51–65
28. Solimanpur M, Vrat P, Shankar R (2004) Ant colony optimization algorithm to the inter-cell layout problem in cellular manufacturing. Eur J Oper Res 157:592–606
29. Wu X, Chu C-H, Wang Y, Yan W (2006) Concurrent design of cellular manufacturing systems: a genetic algorithm approach. Int J Prod Res 44:1217–1241
30. Chan FTS, Lau KW, Chan PLY, Choy KL (2006) Two-stage approach for machine-part grouping and cell layout problems. Robot Comput Integr Manuf 22:217–238
31. Tavakkoli-Moghaddam R, Javadian N, Javadi B, Safaei N (2007) Design of a facility layout problem in cellular manufacturing systems with stochastic demands. Appl Math Comput 184:721–728

32. Mahdavi I, Mahadevan B (2008) CLASS: an algorithm for cellular manufacturing system and layout design using sequence data. Robot Comput Integr Manuf 24:488–497
33. Ahi A, Aryanezhad MB, Ashtiani B, Makui A (2009) A novel approach to determine cell formation, intracellular machine layout and cell layout in the CMS problem based on TOPSIS method. Comput Oper Res 36:1478–1496
34. Ariafar S, Ismail N (2009) An improved algorithm for layout design in cellular manufacturing systems. J Manuf Syst 28:132–139
35. Ma H, Zhang D (2010) The dynamics facility layout study based on cellular manufacturing, 2010 I.E. International Conference on Measuring Technology and Mechatronics Automation pp 862–865
36. Jolai F, Taghipour M, Javadi B (2011) A variable neighborhood binary particle swarm algorithm for cell layout problem. Int J Adv Manuf Technol 55:327–339
37. Leno IJ, Sankar SS, Raj MV, Ponnambalam SG (2011) Bi-criteria optimization in integrated layout design of cellular manufacturing systems using a genetic algorithm, SEMCCO 2011. Lect Notes Comput Sci 7076:323–331
38. Arkat J, Farahani MH, Hosseini L (2011) Integrating cell formation with cellular layout and operations scheduling. Int J Adv Manuf Technol. doi:10.1007/s00170-011-3733-4
39. Kia R, Baboli A, Javadian N, Tavakkoli-Moghaddam R, Kazemi M, Khorrami J (2012) Solving a group layout design model of a dynamic cellular manufacturing system with alternative process routings, lot splitting and flexible reconfiguration by simulated annealing. Comput Oper Res 39:2642–2658
40. Chen CW, Sha DY (1999) A design approach to the multi-objective facility layout problem. Int J Prod Res 37:1175–1196
41. Darwin C (1929) The origin of species by means of natural selection or the preservation of favored races in the struggle for life, New York, The Book League of America (originally published in 1859)
42. Fisher RA (1930) The genetical theory of natural selection. Clarendon, Oxford
43. Holland JH (1975) Adaptation in natural and artificial systems. University of Michigan Press, Ann Arbor
44. Goldberg DE (1989) Genetic algorithms in search optimization & machine learning. Addison Wesley
45. Gen M, Cheng R (2000) Genetic algorithms and engineering optimization. Wiley
46. Aarts E, Korst J (1990) Simulated annealing and the Boltzmann machine. Wiley, New York
47. Mitchell M (1999) An introduction to genetic algorithms. A Bradford Book. The MIT Press, Cambridge
48. Kirkpatrick S, Gelatt CD Jr, Vecchi MP (1983) Optimization by simulated annealing. Science 220:671–680
49. Souilah A (1995) Simulated annealing for manufacturing systems layout design. Eur J Oper Res 82:592–614
50. Balakrishnan J, Cheng CH, Conway DG, Lau CM (2003) A hybrid genetic algorithm for the dynamic plant layout problem. Int J Prod Econ 86:107–120
51. Kuppusamy S (2001) Simulated annealing heuristics for the dynamic facility layout problem, Master's Thesis, West Virginia University
52. Shang J (2002) Ant colony heuristics for the dynamic facility layout problem, Master's Thesis, West Virginia University
53. Singh SP, Sharma RRK (2008) Two-level modified simulated annealing based approach for solving facility layout problem. Int J Prod Res 46(13):3563–3582
54. Kratica J, Tosic D, Filipovic V, Dugosija D (2011) A new genetic representation for quadratic assignment problem. Yugoslav J Oper Res 21(2):225–238