Ant colony optimization algorithm to the inter-cell layout problem in cellular manufacturing

M. Solimanpur [a], P. Vrat [b], R. Shankar [c,*]

[a] *Department of Mechanical Engineering, Indian Institute of Technology, Hauz Khas, New Delhi 110 016, India*
[b] *Indian Institute of Technology, Roorkee 247 667, Uttaranchal, India*
[0] *Department of Management Studies, Indian Institute of Technology, Hauz Khas, New Delhi 110 016, India*

Received 10 July 2002; accepted 14 October 2002
Available online 27 August 2003

Abstract

The inter-cell layout problem is discussed and a mathematical formulation for material flow between the cells is presented. The problem is modeled as a quadratic assignment problem (QAP). An ant algorithm is developed to solve the formulated problem. The performance of the proposed ant algorithm is compared to the facility layout algorithms such as H63, HC63-66, CRAFT and Bubble Search as well as other existing ant colony implementations for QAP such as FANT, HAS-QAP, MMAS-QAP$_{2_opt}$, and ANTS algorithms. The experimental results show that the proposed ant algorithm performs significantly better than the facility layout algorithms. Also, our experimental results reveal that the proposed ant algorithm is effective and efficient as compared to other existing ant algorithms.

Keywords: Facilities planning and design; Cellular manufacturing; Inter-cell layout problem; Quadratic assignment problem; Ant algorithms

1. Introduction

Cellular manufacturing (CM) is a production system in which similar parts are classified into part families (PFs) and dissimilar machines are assigned into machine cells (MCs) in order to exploit the cost-effectiveness of mass production and flexibility of job shop manufacturing. The major advantages of CM have been reported in the literature as reduction in setup time, reduction in throughput time, reduction in work-in-process inventories, reduction in material handling costs, better quality and production control, increment in flexibility, etc. (Wemmerlöv and Hyer, 1989). Minimizing the inter-cellular movements and maximizing the utilization of machines are the two major objectives of interest in the formation and layout of manufacturing cells. In an ideal formation of manufacturing cells, each part is processed in only one cell and no part is transferred from one cell to other. However, in real life applications, there always are some 'exceptional parts' needing

processing on more than one cell. Due to exceptional parts, there would be 'bottleneck machines' that process parts coming from different cell(s). Since exceptional parts are sources of inter-cellular moves, an effective compromise of policies such as subcontracting, duplication of machines and material handling costs are to be made in order to economically manage these items. Some researchers have focused on this issue and developed methods for organizing exceptional parts and bottleneck machines (Shafer et al., 1992; Amirahmadi and Choobineh, 1996; Shankar and Vrat, 1999). As frequently reported in literature, it is almost impractical to achieve an ideal configuration of manufacturing cells with no inter-cellular moves (Urban et al., 2000). This is due to the fact that duplication of bottleneck machines may not be economically justified or physically possible and subcontracting of exceptional parts may not be cost-effective as well. On the other hand, the material handling costs induced by inter-cellular moves is highly affected by the layout of manufacturing cells (Logendran and Chang, 1997). The inter-cell layout problem is an attempt to lay the manufacturing cells so that the total material handling distance or associated cost is minimized.

There are not many researches in the literature addressing the inter-cell layout problem as compared to cell formation problem. Sarker and Yu (1994) developed a two-phase approach for the layout of MCs. In the first phase, the cells are located in such a way that the total material handling cost based on the cell formation solution obtained through a clustering technique is minimized. This stage is followed by the duplication of bottleneck machines in the second phase in order to minimize the total duplication cost and inter-cellular material handling cost. Kouvelis et al. (1995) attempted the single-row layout problem in order to minimize the total backtracking distance. Bazargan-Lari (1999) has explored the application of multi-objective inter-cell and intra-cell layout design methodologies in a CM environment. Non-linear goal programming and simulated annealing approaches have been used to solve this problem. Several inter-cell layout configurations are generated through which a suitable design can be selected. Wang et al. (2001) formulated the facility layout problem in CM environment in which the demand rate of products varies over the products' life cycle. The intra-cell and inter-cell layout problems are simultaneously considered through a non-linear mathematical model using binary variables. The simulated annealing algorithm is further used to minimize the total material handling cost. Wang and Sarker (2002) considered the inter-cell layout problem in CM in which MCs are assigned to different locations so that the total inter-cell material handling cost is minimized. They formulate the inter-cell layout problem as a quadratic assignment problem (QAP) and a three-pair comparison heuristic is developed to solve the QAP. The solution constructed by three-pair comparison heuristic is further optimized through an improvement heuristic called 'Bubble Search'.

QAP is a widely used mathematical model for the facility layout problem (Kusiak and Heragu, 1987). The inter-cell layout problem is one of the applications of QAP in CM. The QAP is a NP-hard optimization problem (Sahni and Gonzalez, 1976). The QAP is addressed as one of the hardest problems that is almost impossible to be optimally solved in an acceptable time for more than 25 facilities/cells. Therefore, several heuristics such as simulated annealing (Connolly, 1990), tabu search (Taillard, 1991), genetic algorithm (Tate and Smith, 1995), scatter search (Cung et al., 1997), ant colony optimization (ACO) algorithms (Maniezzo and Colorni, 1999), etc. have been developed by researchers to provide near-optimal solutions for QAP.

It has been shown that ACO algorithms are among the best performing methods to solve QAP (Stutzle and Dorigo, 1999). There are many implementations of ant algorithms to solve the QAP. Dorigo et al. (1996) applied the ACO algorithm for solving traveling salesman problem (TSP) and then extended their approach to solve asymmetric TSP, QAP and job-shop scheduling problems. In this approach, the heuristic information is defined on the basis of the potential distance and flow of each location and facility, respectively. The primary conjecture in this approach is that good solutions will place facilities with high flow potential on locations with low distance potential. Taillard and Gambardella (1997) proposed a fast ant algorithm namely FANT for QAP. In FANT

algorithm, each iteration uses only one ant and assignments are made without heuristic information. Gambardella et al. (1999) proposed an ant algorithm called HAS-QAP to solve QAP. The major difference of this method as compared to other ant algorithms is that pheromone trails are not used to construct but to modify the solutions. They reported that the HAS-QAP and genetic hybrid (GH) algorithms are among the best methods for solving QAP. Maniezzo and Colorni (1999) improved the performance of the algorithm proposed by Dorigo et al. (1996) and used the lower bound concept to measure the attractiveness of each assignment. In this approach, the lower bound on the objective function value of the partial solution is calculated for each assignment, which indicates its attractiveness. Maniezzo (1999) proposed an interesting ant algorithm to solve QAP, which is referred to as ANTS method in this paper. The major features of ANTS method are: (1) use of lower bounds to measure the attractiveness of assignments, (2) pre-ordering of locations based on the values of dual variables, and (3) use of the moving average of the last K solutions to update the trail levels. The experimental results reported in Maniezzo (1999) indicate the effectiveness of ANTS method. Stutzle and Hoos (2000) proposed a Max-Min ant algorithm called MMAS and used this approach to solve different combinatorial optimization problems. The extension of MMAS algorithm to solve QAP is referred to as MMAS-QAP in this paper. The salient features of this approach are: (1) only one ant is allowed to add pheromone, (2) the range of the trail levels is limited, and (3) the pheromone trails are initialized to the upper trail limit. They showed that MMAS-QAP$_2$_o$_{pt}$ performs well for real life structured problems. A comprehensive review of these implementations is available in Stutzle and Dorigo (1999).

This paper aims to present an effective and efficient ant algorithm for the inter-cell layout problem, which is formulated as QAP. One of the features of ANTS algorithm is the use of lower bounds to obtain the heuristic information of each assignment. Calculation of these lower bounds reduces the efficiency of ANTS algorithm. A new technique based on the partial contribution of each assignment is developed in this paper to alleviate this limitation. The proposed technique is efficient than the lower bound approach used in Maniezzo (1999) and thus saves computation time without deteriorating the quality of solutions. We also address the issue of negative trail levels in the ANTS algorithm and propose a policy to overcome this problem.

This paper is organized as follows: the inter-cell layout problem is discussed in Section 2 and quadratic assignment mathematical model is presented. Section 3 presents a general description of ant colony metaphor applied in optimization theory. The elements of the proposed ant algorithm for inter-cell layout problem are developed in Section 4. Section 5 contains computational results and Section 6 includes conclusions and discussions.

2. Problem statement and formulation

2.1. Notations

The following notations are used throughout this paper:

i, j	indices for cells
m, n	indices for machines
h, l	indices for locations
k	index for ants
p	index for parts
P	total number of parts
D_p	production volume of part p
v	index for moves
C	total number of cells and locations
A	the number of ants in each iteration
/	the total number of iterations
Tabuk	the memory of ant k saving the index of cells already assigned by ant k
V^k	the memory of ant k saving the moves selected by ant k
fij	the frequency of trips from cell i to cell j
dh_l	the distance between locations h and l
a_{ij}^p	$\begin{cases} 1 & \text{if part } p \text{ visits cell } j \text{ immediately after being released from cell } i \\ 0 & \text{otherwise} \end{cases}$

x_{il} $\begin{cases} 1 & \text{if cell } i \text{ is assigned to location l} \\ 0 & \text{otherwise} \end{cases}$

OðlÞ computation complexity of order l

2.2. Material flow among cells

The inter-cell layout problem aims to locate C MCs at C locations so that the expected movement of the material handling device among cells is minimized. It is assumed that the locations are predefined and therefore the distance matrix, D = *[dki]*, is known in which *dhl* represents the distance between locations *h* and l *(h,l = 1; 2;...; C)*. The essential measure in the inter-cell layout problem is to determine the frequency of trips between each pair of cells *i* and *j*. This measure is represented by a flow matrix, F = *\f_tJ]*, in which *fij (i,j= 1;2;...; C)* represents the number of trips from cell *i* to cell *j* in a given time horizon. In the inter-cell layout problem in CMS, the measure *fij* represents the total number of intercellular moves and therefore it mainly depends upon the solution obtained from cell formation problem. In the approach proposed by Wang and Sarker (2002), the frequency of trips from cell *i* to cell *j* is calculated by counting the number of 'ones' that belong to the part family *i* and fall out of diagonal block of cell *j*. This approach has a limitation that it does not consider inter-cell moves which is also decided by production volume and routing of exceptional parts. To overcome this limitation, the following formula is proposed in this paper to calculate the frequency of trips from cell *i* to cell *j*, i.e. *fij*, for a given solution obtained in cell formation phase:

$$f_{ij} = \sum_{p=\backslash}^{P} D_p a_{ij}^p. \qquad (1)$$

As seen in Eq. (1), the proposed formula considers the production volume and the routing of parts to calculate the number of moves from cell *i* to cell *j*. Owing to the fact that some parts visit multiple cells, all parts have been considered in Eq. (1) to calculate the inter-cell transfers from cell *i* to cell *j*. We adopt the same problem considered in Wang and Sarker (2002) to show the calculation of inter-cellular moves using Eq. (1). The configuration given in Wang and Sarker (2002) has been adopted in this paper to make the obtained results comparable. Block diagonal form of this problem along with our superimposed data are shown in Fig. 1 in which eight parts processed by nine machines have been clustered into four cells. A non-zero entry in Fig. 1 implies that the corresponding part requires processing on the related machine and a zero entry means the contrary. Moreover, a non-zero entry shows the sequence at which the related part visits the corresponding machine. The last row in this figure shows the production volumes of parts. Using the Eq. (1) for data given in Fig. 1, the following flow matrix is calculated:

$$\mathbf{F} = \begin{bmatrix} 0 & 155 & 0 & 0 \\ 80 & 0 & 150 & 0 \\ 90 & 0 & 0 & 0 \\ 0 & 0 & 100 & 0 \end{bmatrix}.$$

For example, the material flow from cell 1 to cell 2, f12 = 155, is calculated as follows: the first operation of part 2 is done in cell 1 and then it goes to cell 2 to get processed on machine 1. This transfer creates 80 moves of parts from cell 1 to cell 2 owing to the production volume of part 2. Similarly, part 5 is processed on machine 3 in cell 1 and then is transferred to the home cell, i.e. cell 2, to get processed on machine 1. Consequently, 75 inter-cell moves of parts are created due to part 5. The total number of trips from cell 1 to cell 2 is

		PARTS							
		P_2	P_3	P_5	P_1	P_8	P_4	P_7	P_6
MACHINES	M_1	1	4	1					
	M_8	3	3						
	M_4	2		2	1				
	M_3		1	3	2				
	M_6		2			4	I		
	M_2				3	3			
	M_5					2		2	1
	M_9					I		1	
	M_7							3	2
Production Volume(Dp)		80	90	75	60	100	82	70	68

Fig. 1. The problem adopted from Wang and Sarker (2002) with super-imposed sequence of operations and production volumes of parts.

therefore calculated as *f12* = 80 þ 75 = 155. Two major differences between our approach and the method proposed in Wang and Sarker (2002) are further highlighted as follows:

(1) With respect to part 2, the inter-cellular moves between cell 1 and cell 2 is counted once in Wang and Sarker (2002) whereas it is counted twice in our approach. This is due to the fact that part 2 visits cell 2 in the intermediate stage, i.e. part 2 is initially processed on machine 3 in cell 1 and then goes to cell 2 to get processed on machine 1 and again returns back to cell 1 for its third operation on machine 8. Hence, two inter-cellular moves between cell 1 and cell 2, one move from cell 1 to cell 2 and one move from cell 2 to cell 1, are created for completion of part 2. These moves are multiplied by 80, the production volume of part 2.

(2) With respect to part 3, one inter-cellular move is counted from cell 1 to cell 3 in Wang and Sarker (2002) whereas it is yet possible that no such move is taken place in real situation. As a matter of fact that exact picture will emerge only when sequence of operations is considered. For example for part 3, for a sequence of operations as shown in Fig. 1, no inter-cell moves are indicated from cell 1 to cell 2 and from cell 1 to cell 3. This is due to the fact that part 3 is initially processed in cell 2 and then goes to cell 3 and finally returns to cell 1. If instead of sequence of operations, a zero-one matrix is used like the one by Wang and Sarker (2002), the inter-cell moves from cell 1 to cell 2 is one and from cell 1 to cell 3 is also one.

The differences mentioned above indicate that our proposed approach is more suited to the circumstances encountered in real life problems. Sequence of operations is readily available from the route sheet of parts, which is also the origin of zero-one matrix. The production volume may be obtained from the production plan considering the limitation of capacity of machines.

2.3. Mathematical model

Owing to the assumption that the relative distances between each pair of cells are known a priori, the problem can be mathematically formulated as a QAP. This model is given as follows:

$$\text{Min} \sum_{i=1}^{c}\sum_{7=1}^{C}\sum_{1=1}^{C}\sum_{h=\backslash}^{C} X f j^{dlhxilx} j^{h} \qquad (2)$$

subject to:

$$\sum_{I=1}^{c} r_{il} = 1, \quad i = 1, 2, \ldots, C, \qquad (3)$$

$$\sum_{1=1}^{c} r_{il} = 1, \quad l = 1, 2, \ldots, C, \qquad (4)$$

$$x,_{7}e\{0,1\}, \quad i, l \in \{1, 2, \ldots, C\}. \qquad (5)$$

The objective function (2) minimizes the total material handling distances. The constraint (3) indicates that each location is assigned to only one cell. The constraint (4) ensures that each cell is assigned to only one location. The constraint (5) represents binary decision variables. A complete solution of this problem can be illustrated by a complete permutation of manufacturing cells represented by p. For example, the permutation 7t ð 3;2; 1;4Þ is a feasible and complete solution for an inter-cell layout problem with four cells in which cells 3, 2, 1 and 4 are assigned into locations 1, 2, 3 and 4, respectively.

3. Description of ant algorithms

ACO algorithms or simply ant algorithms are population-based optimization approaches that have been successfully applied to solve different combinatorial optimization problems such as TSP (Dorigo et al., 1996), QAP (Manizezzo and Colorni, 1999), job-shop scheduling (Colorni et al., 1994), etc. ACO algorithms are inspired by the foraging behavior of real life ant colonies in which individual ants deposit a substance called pheromone on the path while moving from the nest to the food sources and vice versa. Thereby, a pheromone trail is formed through which individual ants are aided to smell and select their routes. The paths with higher pheromone would be more likely to be selected by other ants resulting in further amplification of current pheromone trails. Due to this interesting behavior of ants, it is observed that

after some time a colony of ants would select the shortest path from the nest to the food source and vice versa. For more detailed description of this phenomenon, the interested reader is referred to Dorigo et al. (1996).

The attempt in ant algorithms is not to fully simulate the behavior of real ants through mathematical formulations but to derive communication rules among artificial ants. Hence, an artificial ant is a computational agent and an ant algorithm is a colony of agents communicating almost like natural ants. It is in this context that some of the properties outlined for these agents might not be true for real ants. The following characteristics are specifically assumed for artificial ants:

(1) artificial ants are able to memorize their paths,
(2) artificial ants are discarded after each trip from the nest to the food source (each iteration),
(3) artificial ants are not completely blind but able to include heuristic information in their selections and
(4) artificial ants live in an environment where time is discrete.

A generic description of the procedure followed in the ant algorithms is given as follows:

Step 1: Initialize the parameters and pheromone trails.
Step 2: For each ant construct a complete solution.
Step 3: Improve each solution to its local optimum.
Step 4: Update pheromone trails.
Step 5: If continuation is allowed, go to Step 2, otherwise STOP.

4. The proposed ant algorithm

Generally any ant algorithm must specify the following elements:

(1) construction of solutions,
(2) heuristic information,
(3) pheromone updating rule,
(4) selection probability,
(5) local search mechanism, and
(6) termination condition.

Specific implementation of these elements results in distinct ant algorithms with varying degrees of success. There are different implementations of ant algorithms for QAP. In the following subsections we describe our proposed methods for these elements.

4.1. Construction of solutions

A feasible and complete solution of the formulated inter-cell layout problem is considered as a permutation of manufacturing cells. Each part of this solution is termed *state*. Assignment of cell i to location l is called a *move* and represented by $v = ði; lÞ$. A move v takes ant k from state S1 to state S2 and thus the partial solution is gradually completed. Once cell i is assigned to location l in the solution of ant k, the index i is taken into a tabu list denoted by Tabuk. This list is the memory of ant k saving the index of allocated cells. In addition to the list Tabuk, all the moves selected by ant k are stored in a memory represented by V^k. In the next step in which ant k moves from current state to the next state, i.e. another cell is allocated to location $lþ1$, the list Tabuk is used to avoid assignment of cells already allocated. The memory V^k is used at the end of iteration to update the trail levels of moves selected by ants. It is assumed that locations are pre-ordered in an ascending manner. In other words, each ant initially assigns a cell to location 1, then assigns another cell to location 2 and so on till a complete solution is obtained. In each assignment, the corresponding cell number is added to the tabu list of that ant. Thus, the constraints (3)-(5) are satisfied. This leads each ant to generate feasible solution.

4.2. Heuristic information

Unlike the real ants that are almost blind creatures, artificial ants can include some heuristic information while assigning cells at different locations. The heuristic information pertaining to move $v = ði; lÞ$ is denoted by g_{il}. This information indicates the *desirability* of assigning cell i to

location l and is calculated using a heuristic approach. There are several methods to estimate the desirability of each move. Since the heuristic information is calculated for all moves in all ants, it may significantly reduce the efficiency of the algorithm and thus it should be computed in an efficient manner. Let us consider the complexity of this element in the ant algorithms. In the beginning of constructing a solution, each ant has a possibility to assign C cells to location 1, because any cell out of C cells can be allocated to location 1. It is C — 1 possibilities for location 2 and so on. In total, each ant should calculate the heuristic information $C \times (C - 1)/2$ times in each iteration of the algorithm. Thus, the complexity of this element for the entire algorithm becomes $O(I \times A \times C^2)$; where, I is the number of iterations and A is the number of ants in each iteration. Owing to the fact that the number of ants is typically a multiplication of the number of cells, it is concluded that providing heuristic information for each move is computationally crucial for the efficiency of the algorithm.

In the earlier implementations of ant algorithms, the heuristic information is calculated either a priori (Dorigo et al., 1996; Dorigo and Gambardella, 1997) or a posteriori (Maniezzo and Colorni, 1999). In the first category of implementations, the heuristic information is computed first at the beginning and remains unchanged during the running of algorithm. In the second category, the heuristic information is dynamic and depends upon the current state of the ant. Two conflicting aspects are to be considered in the calculation of heuristic information. These are: (1) the efficiency of calculation, and (2) the quality of information. The implementations with a priori heuristic information are efficient but not thoroughly indicating the desirability of moves. On the contrary, in the second category of implementations, a precise estimation of the desirability of each move is obtained, though the efficiency of computation is not satisfactory.

An attempt has been made in this paper to develop a method for calculating heuristic information by a rationale combination of these aspects. The proposed method to calculate the heuristic information falls in the second category, i.e. implementation with a posteriori information. However, the computational complexity of our proposed approach is less than that of some methods falling in this category such as Maniezzo and Colorni (1999). The proposed formula considers the partial contribution of each move to the objective function value. Let p denote a permutation of cells under construction. Since, while calculating the desirability of move $v = (i, l)$ the permutation of cells till position l, i.e. $p(1), p(2), \ldots, p(l-1)$ is known, the partial contribution of move $v = (i, l)$ can therefore be calculated as follows:

partial contribution of move (i, l)

$$= \sum_{s=1} (f_{\pi(s)i} \times d_{sl} + f_{i\pi(s)} \times d_{ls}).$$

It is assumed that the lower is the partial contribution of a move, the higher would be the desirability of that move. Based on this assumption, the following formula is proposed to calculate the desirability of move $v = (i, l)$:

$$\eta_{il} = \frac{1}{1 + \sum_{s=1}^{l-1} (f_{p(s)i} \times d_{p(s)l} + f_{ip(s)} \times d_{lp(s)})}. \qquad (6)$$

The reason due to which number 1 is added in the denominator of (6) is for avoiding division by 0. The second term in the denominator is assumed to be greater than or equal to zero for the inter-cell layout problem as the material flows and the distances across the cells are positive values. Due to the hyperbolic function given in (6), the moves with small augmentation in the objective function would be more desirable for selection. The heuristic information given by Eq. (6) can be obtained in $O(l)$.

4.3. Pheromone updating rule

Real ants deposit a substance called pheromone while moving from one point to another point. Artificial ants perform this action by adding a value called *trace* on the trail levels of moves chosen by them. The updating of trail levels can be performed either after each move or after completion of solution. Dorigo et al. (1996) have experimentally shown that the performance of second way is much better than the first.

A successful mechanism for updating trail levels has been proposed by Maniezzo (1999) in which each solution is evaluated against the last K solutions. The trail levels do not change till the first K solutions are formed. Once formed, the moving average of last K solutions, represented by $\bar{z}$ is computed and the objective function value of new solution, represented by z^k is compared to $\bar{z}$. If zk is lower than $\bar{z}$, the trail levels of the moves selected by ant k are increased, otherwise the trail levels of the same are decreased. One of the major advantages of this approach is the elimination of a sensitive factor in the ant algorithms, namely evaporation rate. Although a new parameter K, which is the cardinality of the moving average is introduced, Maniezzo (1999) reports that it is not as critical as the evaporation rate.

We have modified this mechanism to calculate $\bar{z}$ at the end of each iteration. In other words, in our proposed approach each solution is evaluated against the other solutions obtained in that iteration. This mechanism is more efficient than that of Maniezzo (1999) as trail levels are updated at the end of iterations rather than after completion of each solution. Also, the proposed ant algorithm does not need to introduce the so-called control parameter K as the value $\bar{z}$ is computed using the solutions of each iteration. The pheromone updating mechanism is applied through the following equations:

$$\tau_{il}(t) = \tau_{il}(t-1) + \sum_{k} \Delta\tau_{il}^{k}, \tag{7}$$

$$\text{AT}_{il}^{*} = \text{TO} \times \left(1 - \frac{z^{*} - \text{LB}}{\bar{z} - \text{LB}}\right), \quad \text{Vu} = (i, I) \; e \; V^{k}, \tag{8}$$

where zk is the objective function value of the solution obtained by ant k, $\bar{z}$ is the average of the objective function values of the solutions obtained in iteration t and LB is a lower bound to the optimum value of objective function. The control parameter S0 is a small real number indicating the maximum increment in the trail levels due to one ant.

There are many methods to calculate lower bound (LB) for a general QAP. Many of these methods such as Resende et al. (1995) and Carraresi and Malucelli (1992) are inefficient for large size problems. There is currently no lower bound for QAP, which is both tight and efficient for large size problems (Maniezzo, 1999). One of the well known lower bounds for the QAP, trading a rationale compromise between the quality of LB and the computation time, has been independently reported by Gilmore (1962) and Lawler (1963) which is referred to as the Gilmore and Lawler lower bound or GLB in abbreviation form in literature. Although the GLB does not create a tight lower bound specifically for large size problems (Maniezzo, 1999), it is adopted in this paper as a tighter lower bound is not so important in our proposed approach. The lower bound GLB is obtained through solving the following linear assignment problem:

$$\text{GLB} = \text{Min} \sum_{i=1}^{C} \sum_{l=1}^{C} XXz_{il} \; \text{xx}_{,7} \tag{9}$$

subject to:

$$\sum_{i=1}^{C} xil = 1; \quad l = 1;2;\ldots;C; \tag{10}$$

$$\sum_{l=1}^{C} J> = 1; \quad i=l,2,...,C, \tag{11}$$

$$x_{il} \; 2 \; f0; \; 1g; \quad i; \; l = 1;2; \ldots;C; \tag{12}$$

where zil in Eq. (9) represents the minimum contribution due to assignment of cell i to location l. The $z_{i}l$ coefficients in Eq. (9) can be calculated by arranging the elements of row i of matrix F and row l of matrix D in descending and ascending order, respectively, and then computing their inner product.

4.4. Selection probability

In all the implementations of ant algorithms, an ant chooses amove to go from current state, *S1*, to the next adjacent state, *S2*, based on a rationale combination of two factors, namely the desirability of that move and the trail level of that move. There are different methods in literature to combine these factors. We have modified the method proposed by Maniezzo (1999) to calculate the selection probability of each cell not yet assigned.

There are two reasons for adopting and modifying this method. The first is the simplicity of the approach proposed by Maniezzo (1999) as only one control parameter, i.e. a, is used to map the relative importance of trail level and the desirability of each move. In other methods, however, one parameter is considered for each factor. The second reason is the computational efficiency of this method as multiplication operations are used instead of exponentiations. We explain the genesis of our approach as follows. According to the method proposed by Maniezzo (1999), ant k chooses cell i to assign to location l by the following probability:

$$p_{''}^{k} = \frac{\alpha \times \tau_{il} + (1-\alpha) \times \eta_{il}}{\text{E,y}_{\text{T}}\text{abu}_{\text{t}}\ 0\ \text{x}\ \text{T,}_{7} + (1\ -\ \text{a})\ \text{x}\ r\backslash_{a})'} \qquad i \notin \text{Tabu}_{k}. \tag{13}$$

We call the term a x s_{il} þ ð1 — a) x gi_{l} in Eq. (13) as the *fitness* of cell i for location l. Application of Eqs. (7), (8) and (13) may result in negative trail levels or fitness values. We investigated the performance of two policies in tackling the moves with negative trail levels or fitness values. These are as follows:

(1) Moves with negative trail levels are avoided and the selection is only done amongst the moves with positive values. Moves are chosen with equal chances if there is no move with positive trail level.
(2) The absolute value of the most negative trail level is added to the trail levels of other moves and therefore all the trail levels become positive. Thus, the move possessing the most negative trail level will have no chance for selection.

It is to be noted that the search space in the first policy is more limited than that of in the second. This fact is in agreement with our experimental observations signifying that the second policy functions better than the first. Our proposed method performs as follows: before applying the Eq. (13) for calculating the probability of assigning cell i to location l in the solution being constructed by ant k, the fitness of all allowed moves are calculated. In case of any negative fitness value, the second policy is employed and then the Eq. (13) is used to compute the related probabilities.

4.5. Local search mechanism

All the effective implementations of ant systems are hybrid algorithms employing some kind of local optimization techniques such as 2-opt technique, tabu search, simulated annealing etc. (Stutzle and Dorigo, 1999). Ant algorithms with effective performance mostly operate as follows: ACO meta-heuristic is used to construct an initial solution. Using a local search mechanism the constructed solution is further improved to its local optimum and finally the trail levels are updated on the basis of the improved solution. This integration may significantly increase the effectiveness and the efficiency of ant algorithms. One of the major drawbacks of constructive algorithms proposed for facility layout problems is that these methods are not as effective as improvement algorithms (Kusiak and Heragu, 1987). On the other hand, applying improvement algorithms to improve several randomly generated solutions may not lead to the optimum or nearly optimum solutions as the initial solutions are quite far from optimality. Moreover, since the improvement algorithms start from bad solutions, the computation time required by improvement algorithms would be more. Besides, ant algorithms are population-based adaptive meta-heuristics that are able to construct relatively good solutions and therefore their integration with a local search mechanism may result in optimum or near optimum solutions. Owing to the fact that the quality of initial solutions created by ant algorithms are good, the integrated local search mechanism needs only a few iterations to enrich these solutions to their local optimum resulting in reasonable computation time.

A major concern in the application of hybrid ant algorithms for solving inter-cell layout problem is the selection of an appropriate local search mechanism. Taillard (1995) classifies different instances of QAP into four groups as follows:

(i) First group consists of unstructured, randomly generated instances in which problem data are generated randomly based on uniform distribution.
(ii) Second group consists of unstructured instances with grid distances in which the entries

of distance matrix are defined according to Manhattan distance between grid points and the entries of flow matrix are created randomly.

(iii) Third group contains real-life instances extracted from practical applications. The flow entries of real-life instances are at most equal to zero and the non-zero entries are not uniformly distributed.

(iv) Fourth group includes real-life like instances in which the problem data are generated randomly but resemble the distribution of real-life instances.

Two quantitative measures, viz. *flow dominance* and *distance dominance* have been proposed by Vollmann and Buffa (1966) that can be used to differentiate among these classes of QAP instances. The unstructured randomly generated instances possess low flow dominance whereas structured real-life instances are of higher flow dominance. Empirical findings reported in Stutzle and Dorigo (1999) reveal that for the structured real-life problems a fast local search technique such as 2-opt method would function satisfactorily.

On the other hand, the flow matrix entries in the inter-cell layout problem depend upon the solution obtained through the cell formation phase. Since the inter-cellular moves are minimized in the cell formation problem therefore the entries of the flow matrix in the inter-cell layout problem are mostly equal to zero. Consequently, the inter-cell layout problem would be of real-life type problems for which a simple local search mechanism is good enough.

Due to the reasons mentioned above, a simple deterministic 2-opt algorithm is adopted for the inter-cell layout problem to carry the solutions obtained by ants into their local optimum. In the 2-opt local search algorithm employed in the current research, the objective function values associated with all possible exchanges of cells are evaluated. The exchange with maximum reduction in the objective function is selected and performed on current solution. This process is repeated until the objective function value of current solution can not be reduced further.

4.6. Termination condition

Similar to other meta-heuristics such as genetic algorithm, tabu search, simulated annealing, etc. the ant algorithm can be terminated in several manners, e.g. repeating the algorithm for a maximum number of iterations, running for a stipulated time, etc. Maniezzo (1999) and Maniezzo and Colorni (1999) use a time limit of 10 minutes to end up the running of their ant algorithms for QAP. The empirical observations reported in these researches reveal that the ant algorithm reaches a *stagnation* state in earlier stages of the execution of algorithm. Stagnation is a situation in which all the ants converge into the same solution and therefore the solution can not be further improved. Consequently, we decided to run the algorithm till stagnation situation is encountered. To do so, the best objective function value obtained in each iteration is recorded and compared with that of obtained in earlier iterations. The algorithm stops if the objective function value has not been improved within the last certain number of iterations.

4.7. The ant algorithm

In this subsection, the elements discussed above are synthesized to evolve the proposed ant algorithm. The pseudo code of the proposed ant algorithm is given below:

```
Initialization
1. Set values of parameters A, max_iter,
   oc and s0. Set counter = 0.
2. Initialize the trail levels of all
   moves, sil (i; l = 1,2;...; C).
3. Compute the lower bound, LB for the
   problem.

   Iteration
4. for k = 1 to A
     Set Tabuk = U.
     for l = 1 to C
       for any cell not appeared in
       Tabuk, say i, Compute the gil and
       the probability of assigning
       cell i to location l, i.e.pilk.Use
       the second policy discussed in
```

Section 4.4 to treat the negative trail levels. Use Eq. (13) to assign a cell to location l and store the selected cell in Tabuk.
end
Use the local search mechanism discussed in Section 4.5 to improve the solution constructed by ant k. Save the improved solution in the memory of ant k, V^k.

Compute the average of objective function values of solutions obtained in current iteration.

5. for $k = 1$ to A
   Use Eqs. (7) and (8) to update the trail levels of the moves saved in the memory of ant k, V^k.
6. If the best solution obtained in the last iteration has been improved in current iteration, then record the best solution and set $counter = 0$. Otherwise, set $counter = counter + 1$.
7. If $counter < max\_iter$, go to Step 4. Otherwise, STOP.

5. Computational results

The proposed ant algorithm has been coded in Borland C++ and executed on a 166 MHz Pentium I PC so as to remain comparable in terms of the computational effort of ANTS method. The proposed algorithm contains four parameters A, max_iter, a and S0. These parameters affect the performance of the proposed ant algorithm. Extensive experimental tests were conducted to see the effect of different values on the performance of the proposed algorithm. Based on these observations, the following experimentally derived rules are proposed to set the values of parameters:

$$A = 3xC, \quad \text{maxiter} = C; \quad a = 0.8; \quad s_0 = 0.25;$$

where C is the number of manufacturing cells to be located.

The performance of the proposed ant algorithm is evaluated through two different schemes:

1. comparison of the proposed ant algorithm with other heuristics particularly developed for the facility layout problem,
2. comparison of the proposed ant algorithm with other ant algorithms recently proposed for QAP.

For the first scheme, the eight problems reported in Nugent et al. (1968) have been selected in this paper. Although, these problems are not specifically inter-cell layout problems, these are selected because these have been widely used by other researchers to test the performance of their methods. The solutions of these problems are available for facility layout techniques H63, HC63-66 and CRAFT (Nugent et al., 1968). Along this line, Wang and Sarker (2002) developed an algorithm for the inter-cell layout problem called as Bubble Search and used these problems to evaluate the effectiveness of their algorithm. In Nugent et al. (1968) and Wang and Sarker (2002), the methods H63, HC63-66, CRAFT and Bubble Search (BS) are used to solve these problems five times each, starting from a different solution. On similar lines, we have run the proposed ant algorithm five times to solve these problems. Table 1 shows the average objective function value of each problem obtained over five runs of each algorithm. The second column in this table shows the optimum solution of each problem. The optimum solutions have been put as bold in Table 1. As seen in Table 1, the proposed ant algorithm has obtained the global optimum solution in all but the last problem. Among the five solutions obtained by the proposed algorithm for the last problem (the problem with size 30), four solutions are the global optimum (3062) and one solution is near-optimal (3064). The proposed ant algorithm outperforms all the H63, HC63-66, CRAFT and BS methods in terms of the quality of solution. The last column in Table 1 shows the average computation time of the proposed ant algorithm for each problem. The time recorded as 0 indicates that the computation time is less than hundredth of a second.

The second scheme of our experimentation is to compare the performance of the proposed ant algorithm with other ACO algorithms recently proposed for QAP. As discussed earlier, the inter-cell

Table 1
Comparison of different facility layout methods applied to the problems in Nugent et al. (1968)[a]

Number of cells	Global optimum solution	Algorithms used in Nugent et al. (1968)			The bubble search algorithm used in Wang and Sarker (2002)	The proposed ant algorithm	
		H63	HC63-66	CRAFT		Solution	Time (seconds)
5	**25**	27.6	29.4	28.2	25.2	**25**	0
6	**43**	44.2	44.2	44.2	43	**43**	0
7	**74**	78.8	78.4	79.6	75	**74**	0.1
8	**107**	114.4	110.2	113.4	109	**107**	0.16
12	**289**	317.4	310.2	296.2	301.6	**289**	0.58
15	**575**	632.6	600.2	606	585.8	**575**	1.74
20	**1285**	1400.4	1345	1339	1332.6	**1285**	7.38
30	**3062**	3267.2	3206.8	3189.6	3165.2	3062.4	108.2

[a] The values below the each heuristic are the average objective function value obtained over five runs of the heuristic. The optimum solutions are in boldface.

layout problem is a structured problem (types (iii) and (iv) discussed in Section 4.5) in which a majority of the inter-cellular moves are equal to zero. Therefore, we consider those ant algorithms, which have been shown to perform well for structured problems. Specifically, we compare the performance of the proposed ant algorithm with FANT (Taillard and Gambardella, 1997), HAS-QAP (Gambardella et al., 1999), ANTS (Maniezzo, 1999) and MMAS-QAP$_{2\text{-}o}$pt (Stutzle and Hoos, 2000) methods. In addition, the performance of the proposed algorithm is compared to the GH method proposed by Fleurent and Ferland (1994), which has been reported to perform well for the structured real-life problems (Taillard, 1995).

We have selected all the structured problems in QAPLIB for which 19 6 C 6 30, where C is the size of problem (the number of cells in the inter-cell layout problem). The reason for selecting these problems is that the smaller problems are easy to solve and almost all the ant algorithms obtain the global optimum solution. On the other hand, the problems with size 30 is sufficient for the approach as the number of manufacturing cells in real world applications is generally low. A survey of 32 users of manufacturing cells in various industries of US indicates that the average number of cells implemented by these industries is six (Wemmerlov and Hyer, 1989).

Table 2 shows the quality of solutions for all selected problems obtained through different methods, which is measured as the percentage deviation from the best-known solution. The solutions marked as '*' in this table are the optimum solutions. As seen in this table, both the ANTS method and the proposed ant algorithm either obtain better solutions or perform as good as other algorithms for all the problems. The ANTS method and the proposed ant algorithm obtain the best-known solution in all but one problem. The exception for ANTS algorithm is problem chr25a and for the proposed ant algorithm it is problem kra30b).

While the effectiveness of the ANTS method and the proposed ant algorithm is comparable, comparison of the computation times for these methods reveals that the proposed ant algorithm is computationally efficient than the ANTS method. We have not compared the computation time of the proposed ant algorithm with other methods as: (1) all the considered heuristics but the ANTS method have been executed on mini-computers (GH, HAS-QAP and FANT on Sun Sparc 5 and MMAS-QAP2_opt on Sun UltraSparc I processors) and therefore the related data are not comparable due to differences in computing facilities, and (2) the computation time is not available for some of these heuristics. Table 3 shows the computation time of the proposed ant algorithm and the ANTS method; both obtained on the 166 MHz Pentium I PC and averaged over five runs of these methods for each problem. In this table, there are two values for each method on each problem: (1) the total

Table 2
The quality of solutions obtained by different heuristics for structured QAP instances (types (iii) and (iv)) selected from QAPLIB[a]

Problem name	Size	Best known value[b]	GH	HAS-QAP	MMAS-QAP$_{2\ opt}$	FANT	ANTS	The proposed ant algorithm
els19	19	17212548*	0	0	-	0	-	0
tai20b	20	122455319*	0	0.0905	0	0.0905	0	0
chr25a	25	3796*	2.6923	3.0822	-	–	0.76	0
tai25b	25	344355646*	0	0	0	0	0	0
bur26a	26	5426670*	0.012	0	0	0.0253	0	0
bur26b	26	3817852	0.0219	0	0	0.0171	0	0
bur26c	26	5426795	0	0	0	0	0	0
bur26d	26	3821225	0.0002	0	0	0	0	0
bur26e	26	5386879	0	0	0	0	0	0
bur26f	26	3782044	0	0	0	0	0	0
bur26g	26	10117172	0	0	0	0	0	0
bur26h	26	7098658	0.0003	0	0	0	0	0
kra30a	30	88900*	0.1338	0.6299	0.314	1.0304	0	0
kra30b	30	91420*	0.0536	0.0711	0.049	0.0919	0	0.0153
tai30b	30	6.37E+08	0.0003	0	0	0.0003	0	0

[a] All the values below the heuristics indicate the percentage deviation of the solution obtained by the related heuristic from the best-known solution. The percentage deviations obtained by GH, HAS-QAP and MMAS-QAP$_{2\ opt}$ algorithms have been averaged over ten runs of these algorithms for solving each problem and it has been averaged over the five runs of the ANTS method and the proposed ant algorithm.

[b] The solutions marked as '*' are the global optimum solution.

Table 3
The efficiency of ANTS method and the proposed ant algorithm for solving structured QAP instances (types (iii) and (iv)) selected from QAPLIB[a]

Problem name	Size	ANTS		The proposed ant algorithm		PRACT$_{best}$ (%)
		t_{best} (seconds)	$\overline{T}$ (seconds)	t_{best} (seconds)	$\overline{T}$ (seconds)	
els19	19	–	–	5	13	–
tai20b	20	3	600	1.9	9	36.7
chr25a	25	551	600	33.7	53	93.9
tai25b	25	7	600	5	29.9	28.6
bur26a	26	10	600	3.1	79.7	69
bur26b	26	26	600	4.6	74.5	82.3
bur26c	26	13	600	3.7	79.2	71.5
bur26d	26	48	600	5	75.7	89.6
bur26e	26	12	600	6.6	82.2	45
bur26f	26	20	600	6.6	76	67
bur26g	26	11	600	4.3	79.3	60.9
bur26h	26	17	600	2.3	71.9	86.5
kra30a	30	46	600	24.9	86.1	45.9
kra30b	30	25	600	56.3	111	)125.2
tai30b	30	223	600	70.5	138.8	68.4
Total		1012	8400	233.5	1059.3	720.1
Average		72.3	600	15.6	70.6	51.4

[a] All the times are averaged over the five runs of each algorithm on each problem.

computation time of algorithm $\delta\overline{T}\Phi$, and (2) the time needed to obtain the best solution of each problem (t_{best}). The time $\overline{t}b_{est}$ indicates that how fast the algorithm obtains the final solution of each

problem. Since the ANTS algorithm has been run for 10 minutes (600 seconds) to solve each problem, the total computation time, i.e. $\overline{T}$, would be 600 seconds for the ANTS algorithm in all the problems. As seen in Table 3, the total computation of the proposed ant algorithm is considerably less than ANTS algorithm. Also, the time **tbest** is in favor of the proposed ant algorithm (except in problem kra30b). In the last column of Table 3, we calculate the 'Percentage Reduction in the Average Computation Time' needed to obtain the best solution of each problem ðPRACT$_{best}$Þ% as a measure of the efficiency of the proposed algorithm. This measure is calculated as follows: *a* is the time needed to obtain the best-known solution using the ANTS algorithm, *b* is the time needed to obtain the best-known solution using the proposed algorithm,

$$PRACT_{best}(\%) = 100 * \{a-b)/a.$$

As seen in the last row of Table 3, the proposed ant algorithm has reduced the computation time needed to obtain the best solution of each problem by 51.4% on average.

6. Conclusions and discussions

The inter-cell layout problem addressed in this paper is an important issue in the successful implementation of CM. The problem is mathematically modeled as a QAP. One of the important data required for the layout of manufacturing cells is the flow of materials across different cells. In earlier attempts, it is mostly assumed a priori known data. In this paper, the sequence of operations and the production volume of parts are introduced as two major factors affecting the intercellular flow of materials. A mathematical model is proposed for calculating the material flow among the cells. An ant algorithm is developed to solve the formulated problem. The performance of the proposed algorithm is compared with other heuristics developed for facility layout problem as well as many other ant algorithms recently developed for QAP. The proposed algorithm obtains the optimum solution for all the problems selected from Nugent et al. (1968) and thus outperforms H63, HC63-66, CRAFT and 'Bubble Search' methods in terms of the total material handling distance in all the problems. Also, the proposed ant algorithm either outperforms the GH, FANT, HAS-QAP and MMAS-QAP$_2$^$_p$t algorithms or it performs as good as these methods for the structured real life problems. Based on the conducted experiments, while the effectiveness of the proposed ant algorithm is comparable with ANTS algorithm, it is shown that the proposed ant algorithm is more efficient than the ANTS algorithm in terms of the computation time. However, our conclusion is limited for structured real life problems for which 19 6 *C 6* 30 where *C* is the size of problem. Since the empirical survey of users has indicated that the size of inter-cell layout problem in real world is generally small, the proposed algorithm is suitable for real life problems. The attempted experiments reveal that the proposed ant algorithm is effective and efficient for the inter-cell layout problem. It is reported in the literature that the computation time of ant algorithms can even be significantly reduced by implementation of ant algorithms on parallel processing machines, which needs further exploration.

Acknowledgements

The authors would like to put on record their appreciation to two anonymous referees for their valuable suggestions, which have enhanced the quality of the paper over earlier versions.

References

Amirahmadi, F., Choobineh, F., 1996. Identifying the composition of cellular manufacturing system. International Journal of Production Research 34 (9), 2471-2488.

Bazargan-Lari, M., 1999. Layout designs in cellular manufacturing. European Journal of Operational Research 112, 258-272.

Carraresi, P., Malucelli, F., 1992. A new lower bound for the quadratic assignment problem. Operations Research 40 (supp. 1), S22-S27.

Colorni, A., Dorigo, M., Maniezzo, V., Trubian, M., 1994. Ant system for job-shop scheduling. JORBEL—Belgian Journal of Operations Research Statistics and Computer Science 34 (1), 39-53.

Connolly, D.T., 1990. An improved annealing scheme for the QAP. European Journal of Operational Research 46, 93-100.

Cung, V.-D., Mautor, T., Michelon, P., Tavares, A., 1997. A scatter search based approach for the quadratic assignment problem. In: Baeck, T., Michalewicz, Z., Yao, X. (Eds.), Proceedings of ICEC'97. IEEE Press, pp. 165-170.

Dorigo, M., Gambardella, L.M., 1997. Ant colony system: A cooperative learning approach to the traveling salesman problem. IEEE Transactions on Evolutionary Computation 1 (1), 53-66.

Dorigo, M., Maniezzo, V., Colorni, A., 1996. The ant system: Optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man and Cybernetics 1 (26), 29-41.

Fleurent, C., Ferland, J., 1994. Genetic hybrids for the quadratic assignment problem. DIMACS Series in Mathematics and Theoretical Computer Science 16, 190-206.

Gambardella, L.M., Taillard, E.D., Dorigo, M., 1999. Ant colonies for the QAP. Journal of Operational Research Society 50, 167-176.

Gilmore, P., 1962. Optimal and suboptimal algorithms for the quadratic assignment problem. Journal of SIAM 10, 305-313.

Kouvelis, P., Chiang, W., Yu, G., 1995. Optimal algorithms for row layout problems in automated manufacturing systems. IIE Transactions 27 (1), 99-104.

Kusiak, A., Heragu, S.S., 1987. The facility layout problem. European Journal of Operational Research 29, 229-251.

Lawler, E., 1963. The quadratic assignment problem. Management Science 9, 586-599.

Logendran, R., Chang, S.K., 1997. Manufacturing cell formation in the presence of flexible cell locations and material transporters. Computers and Industrial Engineering 33, 545-548.

Maniezzo, V., 1999. Exact and approximate nondeterministic tree-search procedures for the quadratic assignment problem. INFORMS Journal on Computing 11, 358-369.

Maniezzo, V., Colorni, A., 1999. The ant system applied to the quadratic assignment problem. IEEE Transactions on Knowledge and Data Engineering 11 (5), 769-778.

Nugent, C.E., Vollman, T.E., Ruml, J., 1968. An experimental comparison of techniques for the assignment of facilities to locations. Operations Research 16, 150-173.

Resende, M.G.C., Ramakrishnan, K.G., Drezner, Z., 1995. Computing lower bounds for the quadratic assignment problem with an interior point algorithm for linear programming. Operations Research 43, 781-791.

Sahni, S., Gonzalez, T., 1976. P-complete approximation problems. Journal of ACM 23, 555-565.

Sarker, B.R., Yu, J., 1994. A two-phase procedure for duplicating bottleneck machines in linear layout, cellular manufacturing system. International Journal of Production Research 32 (9), 2049-2066.

Shafer, S.M., Kern, G.M., Wei, J.C., 1992. A mathematical programming approach for dealing with exceptional elements in cellular manufacturing. International Journal of Production Research 30 (5), 1029-1036.

Shankar, R., Vrat, P., 1999. Some design issues in cellular manufacturing using the fuzzy programming approach. International Journal of Production Research 37 (11), 2545-2563.

Stutzle, T., Dorigo, M., 1999. ACO algorithms for the quadratic assignment problem. In: Corne, D., Dorigo, M., Glover, F. (Eds.), New Ideas in Optimization. McGraw-Hill.

Stutzle, T., Hoos, H.H., 2000. MAX-MIN ant system. Future Generation Computer Systems 16, 889-914.

Taillard, E.D., 1991. Robust taboo search for the quadratic assignment problem. Parallel Computing 17, 443^155.

Taillard, E.D, 1995. Comparison of iterative searches for the quadratic assignment problem. Location Science 3, 87-105.

Taillard, E.D., Gambardella, L.M., 1997. Adaptive memories for the quadratic assignment problem, Technical Report IDSIA-87-97, IDSIA, Lugano, Switzerland.

Tate, D.M., Smith, A.E., 1995. A genetic approach to the quadratic assignment problem. Computers and Operations Research 22 (1), 73-83.

Urban, T.L., Chiang, W.-C., Russell, R.A., 2000. The integrated machine allocation and layout problem. International Journal of Production Research 38 (13), 2911-2930.

Vollmann, T.E., Buffa, E.S., 1966. The facilities layout problem in perspective. Management Science 12 (10), 450-468.

Wang, S., Sarker, B.R., 2002. Locating cells with bottleneck machines in cellular manufacturing systems. International Journal of Production Research 40 (2), 403-424.

Wang, T.Y., Wu, K.B., Liu, Y.W., 2001. A simulated annealing algorithm for facility layout problems under variable demand in cellular manufacturing systems. Computers in Industry 46, 181-188.

Wemmerlöv, U., Hyer, N.L., 1989. Cellular manufacturing in the US industry: A survey of users. International Journal of Production Research 27 (9), 1511-1530.