Using genetic algorithm to solve dynamic cell formation problem

V. Deljoo\textsuperscript{a,}\textsuperscript{*,} S.M.J. Mirzapour Al-e-hashem\textsuperscript{b}, F. Deljoo\textsuperscript{c}, M.B. Aryanezhad\textsuperscript{b}

\textsuperscript{a} Department of Industrial Engineering, Bu ali Sina University, Hamedan, Iran
\textsuperscript{b} Department of Industrial Engineering, Iran University of Science and Technology, P.C. 1684613114, Tehran, Iran
\textsuperscript{c} Department of Industrial Engineering, Amirkabir University of Technology, Tehran, Iran

\section*{Article Info}

\textbf{Article history:}
Received 7 February 2008
Received in revised form 26 July 2009
Accepted 31 July 2009
Available online 6 August 2009

\textbf{Keywords:}
Dynamic cell formation
Genetic algorithm
Meta heuristics

\section*{Abstract}

In this paper, solving a cell formation (CF) problem in dynamic condition is going to be discussed using genetic algorithm (GA). Previous models presented in the literature contain some essential errors which will decline their advantageous aspects. In this paper these errors are discussed and a new improved formulation for dynamic cell formation (DCF) problem is presented. Due to the fact that CF is a NP-hard problem, solving the model using classical optimization methods needs a long computational time. Therefore the improved DCF model is solved using a proposed GA and the results are compared with the optimal solution and the efficiency of the proposed algorithm is discussed and verified.

© 2009 Published by Elsevier Inc.

1. Introduction

Cell formation (CF) is a part of cellular manufacturing system (CMS) that is actually the implementation of group technology in manufacturing and production systems with the goal of classifying parts in a way that the physical or operational similarities of the parts are used in different aspects of design and production of parts. The advantages derived from cellular manufacturing in comparison with traditional manufacturing systems in terms of system performance have been discussed in Fry et al. [1], Collet and Spicer [2], Levasseur et al. [3], Singh and Rajamaani [4]. These benefits have been established through simulation studies, analytical studies, surveys and actual implementations. They can be summarized as follows:

1. Setup time is reduced.
2. Lot sizes are reduced.
3. Work-in-process (WIP) and finished goods inventories are reduced.
4. Throughput times are reduced.
5. Working flexibility improved.

There is a great amount of literature for cell formation problem ranged from simple models to much extended ones, which attempt to present their models as a tool to take advantages of cell formation as much as possible. For instance, Onwubolu and Mutigi [5] solved a cell formation problem which simultaneously groups machines and part families into cells so that intercellular movements are minimized. Xambre and Vilarinho [6] presented a model for the cell formation problem with multiple identical machines, to minimize intercellular flow.

Dynamic production requirements imply multiple periods when designing a CMS. In this case, the entire planning horizon is divided into multiple periods according to the differences in product mix and/or demand in each period. In each period, product mixes and demands can be deterministic or stochastic [7]. In the case of deterministic product mixes and demands,
they are known in each period. When they are stochastic, the possible product mixes and demands in each period are known with certain probabilities. Literatures in the design of CMSs are reviewed according to the above classifications which are as follows:

1. Design of CMSs for dynamic and deterministic production requirements [8];
2. Design of CMSs for static and stochastic production requirements;
3. Design of CMSs for dynamic and stochastic production requirements [9].

In this research, dynamic and deterministic demands and production requirement condition are discussed.

2. Dynamic and deterministic production requirements

Song and Hitomi [10] developed a method to design flexible manufacturing cells. The method integrates production planning and cellular layout by a long-run planning horizon. The integrated planning model is formulated as a mixed-integer problem which contains two types of integer programming problems: determining the production quantity for each product and the timing of adjusting the cellular layout in a finite planning horizon period with dynamic demand. Chen [11] developed a mathematical programming model for system reconfiguration in a dynamic cellular manufacturing environment. A mixed integer programming model is developed to minimize inter-cell material and machine costs as well as reconfiguration cost in a dynamic cellular manufacturing environment with anticipated changes of demand or production process for multiple time periods. Wicks [12] proposed a multi-period formation of the part family and machine cell formation problem. The dynamic nature of production environment is addressed by considering a multi-period forecast of the product mix and resource availability during the formation of part families and machine cells. The design objectives are the minimization of inter-cell material handling cost, the minimization of investment in additional machines, and the minimization of the cost of system reconfiguration over the planning horizon. Mungwattana [7] proposed a new model for dynamic cell formation problem considering routing flexibility. The planning horizon divided into some periods. Demand and production requirement are given in each period and can be different from one period to another. In addition each operation can be produced in some machine with the different process times and production cost. Mungwattana [7] firstly, presented a nonlinear mixed integer programming and then with some changes, converted it to a linear model and finally it is solved using simulated annealing (SA). Tavakkoli et al. [13] modified the model which is introduced by Mungwattana [7] and presented three Meta heuristic methods consist of GA, SA, and Tabu Search (TS). Then they compared their efficiencies with each other and also with LINGO. Safaei et al. [14] developed an extended model for DCF problem, considering the batch inter/intra cell material handling by assuming the sequence of operations. They also presented a hybrid simulated annealing algorithm to solve their model. Defersha and Chen [15] presented a comprehensive mathematical model for DCF and a two phase GA-based heuristic to solve it. The model attempts to minimize machine investment cost, intercellular material handling cost, operating cost, subcontracting cost, tool consumption cost, setup cost and system reconfiguration cost in an integrated manner. Mehrabad and Safaei [16] proposed a nonlinear DCF model which is a modified version of the model that is presented by Mungwattana [7]. Then they applied neural approach to solve this model. Bajestani et al. [17] proposed a model for DCF problem based on the model which is presented by Mehrabad and Safaei [16]. The model is a multi-objective one and a multi-objective scatter search is used to solve it. Safaei et al. [18] developed a fuzzy model for dynamic cell formation problem which is mainly derived from the model presented by Mungwattana [7] and reformulated by a fuzzy approach. In this model, the fluctuation in part demands and the availability of manufacturing facilities in each period are considered as fuzzy variables. Ahkioon et al. [19] proposed an extended DCF model which considers production planning and system reconfiguration simultaneously to determine optimal cellular layout under deterministic demands of a mixture of products within several planning horizons. Tavakkoli-moghaddam et al. [20] presented a DCF model which is elementally another version of Mehrabad and Safaei [16] with some partial changes. Then they used a simulated annealing to solve their model. Aryanezhad et al. [21] extended DCF model which is presented by Safaei et al. [18] to consider cell formation and worker assignment problem simultaneously. This model is mainly focused on worker assignment and attempts to prove the importance of considering it in dynamic cell formation.

In some previous models for DCF problem, there are some shortcomings in calculation of machine relocation cost and machine purchasing cost. Some attempts are done to remove these shortcomings through these papers [7,13,14]. Mehrabad and Safaei [16,20] which mainly have been less successful. On the other hand, some extended models are developed based on these DCF models and consequently contain these shortcomings as a weak point. Some other models are developed in the literature which was relatively successful to remove these shortcomings [15,19,21].

We studied previous models and reconsidered the shortcomings related to machine relocation cost and machine purchasing cost and some solutions are introduced to solve them. Then, we developed a base model for dynamic cell formation which tries to resolve these two shortcomings using the best formulations exist in different papers in the literature. The goodness of proposed formulation is proved by some reasonable explanations. Furthermore, as DCF is NP-Hard, a new GA is developed to solve this model and its performance is verified.

In the following sections, firstly some literature of GA and its application in CF is reviewed and then the improved DCF model is described. After that proposed GA for solving presented DCF model is explained and its performance is discussed and finally some areas of work are proposed for future researches.
3. Genetic algorithm

The GA is a stochastic searching technique. It can explore the solution space by using the concept taken from natural genetics and evolution theory. In spite of other stochastic search methods, GA searches the feasibility space by setting the feasibility solutions simultaneously in order to find optimal or near-optimal solutions. The procedure is carried out by the use of genetic operations. GA was developed initially by Holland [22]. Goldberg [23] gave an interesting survey of some practical works carried out in this area. Among these, early applications of GA were those developed by Bagley [24] for a game-playing program, by Rosenberg [25] in simulation biology processes and by Cavicchio [26] for solving pattern-recognition problems. Gupta et al. [27] developed a GA for Minimizing total inter-cell and intra-cell movements in cellular manufacturing system. Joines et al. [28] designed an integer-programming model using GA. Venugopal [29] used GA to the machine component-grouping problem with multiple objectives. Tavakkoli et al. [13] developed a GA with matrix chromosomes to solve DCF problem.

4. Problem description

There are some parts in the manufacturing system that each one has some operations which can be done on different machines with different process times (routing flexibility). Part demand for some succeeding periods is known and Parts can move from one machine type to another, according to their flexible routing. New machines can be purchased and added to the cells or some of existing machines can be relocated from one cell to another between periods (machine flexibility). We decided to find the best design of manufacturing cells in each period to optimize the cost of cell formation to satisfy these demands during planning horizon.

In order to explain with more details, an exemplar condition is considered where there are four machine types as follows:

Machine type 1: turning machine.
Machine type 2: milling machine.
Machine type 3: welding machine.
Machine type 4: sawing machine.

And there are four parts with some operations as follows:

(1) Part 1 with 3 operations.
(2) Part 2 with 3 operations.
(3) Part 3 with 3 operations.
(4) Part 4 with 2 operations.

Routing flexibility matrix for doing these operations on different machine types is presented in Table 1:

General schema for this exemplar manufacturing system is shown in Fig. 1:

Previous models presented in the literature contain some essential errors which will decline their advantageous aspects that are improved in the proposed model. In the following sections, this improved model is presented with more details.

4.1. Assumptions

There are some assumptions for DCF which are derived from Tavakkoli et al. [13] and are as follows:

Table 1
Routing flexibility matrix for exemplar condition.

<table>
<thead>
<tr>
<th>Machine type (part, operation)</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1,1)</td>
<td>0</td>
<td>0.3</td>
<td>0</td>
<td>0.5</td>
</tr>
<tr>
<td>(1,2)</td>
<td>0.2</td>
<td>0.4</td>
<td>0.3</td>
<td>0.4</td>
</tr>
<tr>
<td>(1,3)</td>
<td>0</td>
<td>0.2</td>
<td>0</td>
<td>0.3</td>
</tr>
<tr>
<td>(2,1)</td>
<td>0.4</td>
<td>0.3</td>
<td>0.2</td>
<td>0.3</td>
</tr>
<tr>
<td>(2,2)</td>
<td>0</td>
<td>0.3</td>
<td>0.2</td>
<td>0</td>
</tr>
<tr>
<td>(2,3)</td>
<td>0.5</td>
<td>0</td>
<td>0.4</td>
<td>0.3</td>
</tr>
<tr>
<td>(3,1)</td>
<td>0.4</td>
<td>0.3</td>
<td>0.5</td>
<td>0</td>
</tr>
<tr>
<td>(3,2)</td>
<td>0.4</td>
<td>0.3</td>
<td>0</td>
<td>0.5</td>
</tr>
<tr>
<td>(3,3)</td>
<td>0</td>
<td>0</td>
<td>0.2</td>
<td>0.3</td>
</tr>
<tr>
<td>(4,1)</td>
<td>0</td>
<td>0.2</td>
<td>0.3</td>
<td>0.25</td>
</tr>
<tr>
<td>(4,2)</td>
<td>0.4</td>
<td>0.3</td>
<td>0</td>
<td>0*</td>
</tr>
</tbody>
</table>

Values in the table are operation’s process time on machine types (time unit i.e. minute).

* It means that doing this operation on machine type 4 is impossible.
1. The operating times for all part type operations on different machine types are known.
2. The demand for each part type in each period is known.
3. The capability and capacity of each machine type are known and constant over time.
4. Investment or purchase cost per period to procure one machine of each type is known.
5. Operating cost of each machine type per hour is known.
6. Parts are moved between cells in batches. The inter-cell material handling cost per batch between cells is known and constant (independent of quantity of cells).
7. The number of cells used should be specified in advance and it remains constant over time.
8. Bounds and quantity of machines in each cell need to be specified in advance and remain constant over time.
9. Machine relocation from one cell to another is performed between periods and it requires zero time.
10. The machine relocation cost of each machine type is known and it is independent of where the machines are actually being relocated.
11. Each machine type can perform one or more operations (machine flexibility). Likewise, each operation can be done on one machine type with different times (routing flexibility).
12. Inter-cell handling costs are constant for all moves regardless of the distance traveled.
13. No inventory is considered.
14. Setup times are not considered.
15. Backorders are not allowed.
16. No queuing in production is allowed.
17. Machine breakdowns are not considered.
18. Processing capabilities are 100% reliable (i.e. no rework/scrap).
19. Batch size is constant for all productions in all periods.

Fig. 1. General schema for Dynamic cell formation for an exemplar manufacturing system
20. Machines are always available at the beginning of each period (zero installation time).
21. The time value of money is not considered.

4.2. Objective functions

Multiple costs are considered in the design of objective function in an integrated manner. All costs involved in the design of DCF are incorporated; however, it is not possible to consider all costs in the model due to the complexity and computational time required. In this paper, costs are limited to those, which are also related to dynamic and stochastic production conditions and the use of routing and machine flexibility. The objective is to minimize the sum of the following costs:

1. **Machine cost**: The investment and amortization cost per period to procure machines. This cost is calculated based on the number of machines of each type used in the DCF for a specific period.
2. **Operating cost**: The cost of operating machines for producing parts. This cost depends on the cost of operating each machine type per hour and the number of hours required for each machine type.
3. **Inter-cell material handling costs**: The cost of transferring parts between cells, when parts cannot be produced completely by a machine type or in a single cell. This cost is incurred, when batches of parts have to be transferred between cells. Inter-cell moves decrease the efficiency of CM by complicating production control and increasing material handling requirements and flow time.
4. **Machine relocation cost**: The cost of relocating machines from one cell to another between periods. In dynamic and stochastic production conditions, the best CF design for one period may not be an efficient design for subsequent periods. By rearranging the manufacturing cells, the CF can continue operating efficiently as the product mix and demand change. However, there are some drawbacks with the rearrangement of manufacturing cells. Moving machines from cell to cell requires effort and can lead to the disruption of production.

4.3. Input parameters

\[ P \quad \text{number of part types} \]
\[ H \quad \text{number of periods in planning horizon} \]
\[ M \quad \text{number of machine types} \]
\[ C \quad \text{number of cells} \]
\[ OP(p) \quad \text{number of operations should be done on part } p \]
\[ T_{jpm} \quad \text{time required to perform operation } j \text{ of part type } p \text{ on machine type } m \]
\[ R \quad \text{a very large number} \]
\[ D_{ph} \quad \text{demand for product } p \text{ in period } h \]
\[ FN_{m,c} \quad \text{number of machines of type } m \text{ initially exists in cell } c \]
\[ B \quad \text{batch size for inter-cell material handling} \]
\[ x_m \quad \text{purchase cost of machine type } m \text{ (fright cost is considered here)} \]
\[ B_m \quad \text{operating cost per hour of machine type } m \]
\[ \gamma \quad \text{inter-cell material handling cost per batch} \]
\[ INS_m \quad \text{installing cost of machine type } m \]
\[ UNS_m \quad \text{removing cost of machine type } m \]
\[ MT_m \quad \text{capacity of each machine type } m \text{ (h)} \]
\[ LB \quad \text{lower bound cell size} \]
\[ UB \quad \text{upper bound cell size} \]
\[ a_{jpm} \quad \begin{cases} 1 & \text{if operation } j \text{ of part } p \text{ can be done on machine type } m; \\ 0 & \text{otherwise} \end{cases} \]

4.4. Decision variables

\[ N_{m,c,h} \quad \text{number of machines of type } m \text{ used in cell } c \text{ during period } h \]
\[ K_{m,c,h} \quad \text{number of machines of type } m \text{ added in cell } c \text{ at the beginning of period } h \]
\[ K_{m,c,h}^{\text{removed}} \quad \text{number of machines of type } m \text{ removed from cell } c \text{ at the beginning of period } h \]
\[ X_{jpmch} \quad \begin{cases} 1 & \text{if operation } j \text{ of part type } p \text{ is done on machine type } m \text{ in cell } c \text{ in period } h; \\ 0 & \text{otherwise} \end{cases} \]
\[ Z_{jpch} \quad \begin{cases} 1 & \text{if operation } j \text{ of part type is done in cell } c \text{ in period } h; \\ 0 & \text{otherwise} \end{cases} \]

4.5. Mathematical formulation

The mathematical formulation of the proposed DCF model is as follows:
4.6. Objective functions

The objective function consists of several cost items as follows:

(1) **Machine cost:** The investment and amortization cost per period to procure machines. This objective function is non-linear and tries to minimize the cost of purchasing machines during planning horizon. So the real number of machines which are purchased is:

\[
\text{Max}\left\{ 0, \text{Max}_{h} \left\{ \sum_{c=1}^{C} (N_{mch} - FN_{m,c}) \right\} \right\}
\]

This expression shows the number of machines which are purchased during planning horizon and added to the group of machines. To convert this part of objective function to a linear one, tow constraints should be added to the model:

\[
\text{Maxim}_{m,h} \geq \sum_{c=1}^{C} (N_{mch} - FN_{m,c})
\]

\[
Purm \geq \text{Maxim}_{m,h}
\]

where Maxim\(_{m,h}\) presents the maximum number of machines of type \(m\) which are entered to the cellular system during period \(h\) and Purm is maximum number of machines of type \(m\) which are entered to this system on the whole planning horizon. According to these descriptions the objective function should be replaced by this expression:

\[
\sum_{m=1}^{M} x_{m} \cdot \text{Purm}
\]

(2) **Operating cost:** The cost of operating machines for producing parts. This cost depends on the cost of operating each machine type per hour and the number of hours required for them. The cost is incurred by processing operation \(j\) of part \(p\) on machine \(m\) in cell \(c\) during period \(h\) is:

\[
D_{ph} \times T_{ipm} \times X_{ipmch} \times B_{m}
\]
Thus operating cost for all of the operations of products is calculated as follows:

\[ \sum_{h=1}^{H} \sum_{c=1}^{C} \sum_{m=1}^{M} \sum_{p=1}^{P} \sum_{j=1}^{Q(p)} D_{ph} \times T_{jpm} \times X_{jpmch} \times B_m. \]  

(3) **Inter-cell material handling costs:** The cost of transferring parts between cells, when parts cannot be produced completely by a machine type or in a single cell. This cost is incurred, when batches of parts have to be transferred between cells. Inter-cell moves decrease the efficiency of CM by complicating production control and increasing material handling requirements and flow time.

\[ \frac{1}{2} \sum_{h=1}^{H} \left( \frac{D_{ph}}{B} \right) \times \sum_{j=1}^{Q(p)-1} \sum_{c=1}^{C} \sum_{m=1}^{M} \gamma |Z_{j+1}| \times |Z_{j}| \times |Z_{j+1}| \times |Z_{j}|. \]

This is a nonlinear integer equation because of the absolute terms. To transform it into a linear mathematical model, non-negative variables \( y_{jpmch} \) and \( y_{jpmch} \) are introduced and allowed reformulation into a linear model as follows. In the linear model, the objective function is rewritten as:

\[ \frac{1}{2} \sum_{h=1}^{H} \left( \frac{D_{ph}}{B} \right) \times \sum_{j=1}^{Q(p)-1} \sum_{c=1}^{C} \sum_{m=1}^{M} \gamma \times (y_{jpmch} + y_{jpmch}). \]

Considering the following set of constraints:

\[ Z_{j+1, p, c, h} - Z_{j, p, c, h} = y_{jpmch} - y_{jpmch}. \]

(4), (5) **Machine relocation cost:** The cost of relocating machines from one cell to another between periods. It is assumed that when a machine is relocated between cells, it is removed from a cell and transferred to another cell and installed there. In other side if a new machine is purchased, it is merely installed in a cell and fright cost is considered as part of its purchase cost. Machine removal and transference costs are considered jointly by UNS as one of input parameters. Then the sum of machine removal and transference costs for each solution can be calculated as follows:

\[ \sum_{h=1}^{H} \sum_{m=1}^{M} \sum_{c=1}^{C} K_{m, c, h} \times UNS_m. \]

Machine installation cost is considered by INS as one of input parameters. Then the sum of machine installation cost for each solution can be calculated as follows:

\[ \sum_{h=1}^{H} \sum_{m=1}^{M} \sum_{c=1}^{C} K_{m, c, h} \times INS_m. \]

4.7. **Constraints**

Constraints (6) and (7) ensure that each part operation is assigned to one machine and one cell. Constraint (8) ensures that machines capacities are not exceeded and can satisfy the demand. Constraints (9) and (10) specify the lower and upper bounds of cells. Constraints (11) and (12) ensures that the number of machines in the current period is equal to the number of machines in the previous plus the number of machines being moved in and minus the number of machines being moved out. In other words, they ensure conservation of machines over the horizon. In constraint (13) if at least one of the operations of part \( p \) is processed in cell \( c \) in period \( h \), then the value of \( z_{jpmch} \) is equal to one; otherwise it is equal to zero.

5. **Improved DCF model against previous models**

A mathematical model for DCF problem is presented by Mungwattana [7] and in recent years many papers are developed according to this model [13,14,18]. Some shortcomings are observed in these models which should be modified to increase its usefulness in practical applications. In this section, we will discuss these shortcomings and prove our claims about them.

5.1. **DCF model of Mungwattana [7]**

Initially the DCF model which is proposed by Mungwattana [7] is presented as follows:

5.1.1. **Input parameters**

All of decision variables and most of input parameters which is used in this model are the same as SDCWP. Only one different input parameter is used here:
\( \delta_m \) relocation cost of machine type \( m \) which means that cost of removing it from one cell and installing it in another cell.

### 5.1.2. Mathematical formulation

\[
\begin{align*}
\text{Min} & \quad \sum_{h=1}^{H} \sum_{m=1}^{M} \sum_{c=1}^{C} 2^c_m \times N_{m,c,h} + \sum_{h=1}^{H} \sum_{c=1}^{C} \sum_{m=1}^{M} \sum_{p=1}^{P} D_{ph} \times T_{j,p,m} \times X_{j,p,m,c,h} \times B_m + \frac{1}{2} \sum_{p=1}^{P} \sum_{h=1}^{H} F_{ph} \\
& \sum_{j=1}^{Q(p-1)} \sum_{m=1}^{M} \sum_{c=1}^{C} \gamma_m[X_{j+1,p,m,c,h} - X_{j,p,m,c,h}] + \sum_{h=1}^{H} \sum_{m=1}^{M} \sum_{c=1}^{C} (K_{m,c,h}^+ - K_{m,c,h}^-) \times \delta_m.
\end{align*}
\]

(14)

\[
\text{Subject to}
\[
\begin{align*}
\sum_{c=1}^{C} \sum_{m=1}^{M} a_{pm} X_{pmch} &= 1 \quad \forall j, p, h, \\
\sum_{p=1}^{P} \sum_{j=1}^{Q(p)} D_{ph} \times T_{j,p,m} \times X_{j,p,m,c,h} &\leq MT_m \times N_{m,c,h} \forall m, c, h, \\
\sum_{m=1}^{M} N_{m,c,h} + K_{m,c,h}^+ - K_{m,c,h}^- &\geq \text{LB} \quad \forall c, h, \\
\sum_{m=1}^{M} N_{m,c,h} + K_{m,c,h}^+ - K_{m,c,h}^- &\leq \text{UB} \quad \forall c, h, \\
N_{m,c,h-1} + K_{m,c,h}^+ - K_{m,c,h}^- &= N_{m,c,h} \quad \forall m, c, h, \\
N_{mch} \cdot K_{mch}^+ \cdot K_{mch}^- &\geq 0. \text{ Integer}, \\
X_{pmch} &= 0 \text{ or } 1.
\end{align*}
\]

(15)

(16)

(17)

### 5.1.3. Model shortcomings

We will claim here some of shortcomings of this model and prove our claims by exemplar proofs:

1. This model calculates machine relocation cost erroneously. For example assume the following condition for a solution:

\[
\sum_{c=1}^{C} K_{m,c,h}^+ = 5 \forall h, m.
\]

\[
\sum_{c=1}^{C} K_{m,c,h}^- = 4 \forall h, m.
\]

Since the real number of machines type \( m \) is relocated among cells during period \( h \) is equal to 4 (because one of them is purchased not relocated), the real cost for this condition is equal to:

\[
\sum_{c=1}^{C} K_{m,c,h}^- \times \delta_m = 4 \times \delta_m \quad \forall h, m.
\]

But in DCF model this cost is calculated as:

\[
\sum_{c=1}^{C} (K_{m,c,h}^+ + K_{m,c,h}^-) \times \delta_m = (4 + 5) \times \delta_m \quad \forall h, m.
\]

This simple example proves our claim about the miscalculation of relocation cost in DCF model which is presented by Mungwattana [7].

2. This model calculates machine purchasing cost more than its real value. For example assume that in a solution we have encountered by condition as follows:

\[
\sum_{c=1}^{C} N_{m,c,h-1} = 10, \sum_{c=1}^{C} N_{m,c,h} = 11, \sum_{c=1}^{C} N_{m,c,h+1} = 10.
\]

In this example the real number of machines which are purchased during these three periods is 11 units, but according to Eq. (1) this cost is calculated for 31 machines. Considering the heavy expenditures in purchasing ma-
machines in comparison with other source of cost in cell formation, it seems that machine purchasing cost has a noticeable role in this model. And it is necessary to find a better presentation of this cost in the model.

(3) In this model, inter-cell movements is calculated according to the assignment of operations to machines, regardless of the cells. For example, if an operation of one part is assigned to one machine type and the next operation is assigned to another machine type, a material handling cost is incurred and it is independent of the cells in which these jobs are processed while cellular manufacturing, intercell material handling cost is the cost of material movement between cells. Thus the material handling cost calculated by this model is not the real cost of that and it is important because no assumption is used there to clarify the modelers approach.

5.2. Improvement of DCF model in the literature

Some efforts are done in the literature to resolve aforementioned shortcomings in the original DCF model, which are discussed in this section:

(1) To resolve miscalculation of machine relocation cost:
- Tavakkoli et al. [13] changed DCF model of Mungwattana [7] to improve the miscalculation of relocation cost. They changed relocation cost in objective function by \( \sum_{h=1}^{H} \sum_{m=1}^{M} \sum_{c=1}^{C} R_{m,h} \times \delta_m \) where \( R_{m,h} \) was defined as:

\[
R_{m,h} = \min \left\{ \frac{C}{c=1} K_{m,c,h}^+ + \frac{C}{c=1} K_{m,c,h}^- \right\}
\]

For linearization of this definition two constraints was added to this model which are as follows:

\[
R_{m,h} \leq \sum_{c=1}^{C} K_{m,c,h}^+ \quad \forall m, h, \tag{23}
\]

\[
R_{m,h} \leq \sum_{c=1}^{C} K_{m,c,h}^- \quad \forall m, h. \tag{24}
\]

It is easy to show that in this model relocation cost will be zero all of the times: Objective function tries to minimize \( \sum_{h=1}^{H} \sum_{m=1}^{M} \sum_{c=1}^{C} R_{m,h} \times \delta_m \) and it means that the best value for \( R_{m,h} \) is zero. If \( R_{m,h} = 0 \) then constraints (23) and (24) for all of values of \( K_{m,c,h}^+ \) and \( K_{m,c,h}^- \) are always satisfied. Thus the best value for relocation cost always equals zero.
- Safaei et al. [14], Mehrabadi and Safaei [16], Bajestani et al. [17], Safaei et al. [18] and Tavakkoli-moghaddam et al. [20], changed this part of objective function by

\[
\frac{1}{2} \sum_{h=1}^{H} \sum_{m=1}^{M} \sum_{c=1}^{C} (K_{m,c,h}^+ + K_{m,c,h}^-) \times \delta_m \quad \forall h, m. \tag{25}
\]

According to this formulation \( K_{m,c,h}^+ + K_{m,c,h}^- \) cannot be greater than zero concurrently. For example, for a specific problem, let’s assume this result which is obtained for a solution:

\[
\sum_{c=1}^{C} K_{m,c,h}^+ = 5, \sum_{c=1}^{C} K_{m,c,h}^- = 3 \quad \forall h, m.
\]

In this condition, only three machine of type \( m \) is relocated among cells (removed from one cell and installed in another one) and two machines of type \( m \) are purchased and installed in cells. But relocation cost is calculated via Eq. (13) as:

\[
\frac{1}{2} \sum_{h=1}^{H} \sum_{m=1}^{M} \sum_{c=1}^{C} (K_{m,c,h}^+ + K_{m,c,h}^-) \times \delta_m = 4 \times \delta_m.
\]

While real relocation cost is:

\[
\sum_{c=1}^{C} K_{m,c,h} \times \delta_m = 3 \times \delta_m \quad \forall h, m.
\]
- Akhkon et al. [19] used the same method to calculate machine relocation cost but just with different notations.
- Defersa and Chen [15] introduced two new variable terms representing installation and removal costs for each machine type. In fact, relocation cost is divided into installation and removal cost where the former is incurred when a machine which is purchased or transferred from other cells is installed in a cell and the latter is incurred when a machine is removed from cell to be transferred somewhere else. This is a good solution which is
introduced independently by Aryanezhad et al. [21], but just with different notations. Since notations used by
Aryanezhad et al. [21] are more matched with DCF model of Mungwattana [7], we used their formulation in
our proposed model.

(2) To resolve miscalculation of machine purchasing cost:
• Some efforts in the literature such as Tavakkoli et al. [13], Safaei et al. [14], Mehrabad and Safaei [16], Bajestani et al. [17], Safaei et al. [18], and, Tavakkoli-moghaddam et al. [20], used a similar formulation as that of Mungwattana [7]. Miscalculation of machine purchasing cost in their models is discussed in Section 6.1.1.
• Ahkioon et al. [19] used the same method to calculate machine purchasing cost but just with different notations.
• Defersha and Chen [15] calculate purchasing cost of each machine type in a period by multiplying the difference between total number of that machine type in all of the cells in that period and that of previous period, to the purchasing cost of that machine type. This formulation is more correct than other formulations used in previous models, but similar to preceding ones, initial pattern of cells is not regarded here. On the other hand it is reasonable to store redundant machines somewhere to be used in next periods, which is ignored here too.

In each period if the number of machines from one type in one cell is greater than what is needed, they are removed from that cell and installed in the other cells in which they are needed (if any). Otherwise they are stored somewhere. In the succeeding periods, if additional machines are needed, they are provided firstly from the store (if any). So the real number of machines which are purchased is:

$$\text{Max} \left\{ 0, \text{Max}_{m,h} \left( \sum_{c=1}^{C} (N_{\text{mach}} - FN_{m,c}) \right) \right\}.$$ 

This is used as a part of objective function in the proposed model. Variable $FN_{m,c}$ which presents number of machine type $m$ initially exists in cell $c$, is introduced to consider initial pattern of cells.

(3) To improve miscalculation of inter-cell material handling cost:
• Tavakkoli et al. [13] improved this shortcoming by introducing a new binary variable:

$$Z_{j,p,c,h} = \begin{cases} 1 & \text{if operation } j \text{ of part type } p \text{ is done in cell } c \text{ in period } h, \\ 0 & \text{otherwise}. \end{cases}$$

This is calculated using a new constraint as follows:

$$\sum_{m=1}^{M} X_{j,p,m,c,h} = Z_{j,p,c,h} \quad \forall j, p, c, h, \quad (26)$$

and finally inter-cell material handling cost is calculated as follows:

$$\frac{1}{2} \sum_{h=1}^{H} \frac{D_{p,h}}{B} \times \sum_{j=1}^{O_{p} - 1} \sum_{m=1}^{M} \gamma |Z_{j,p,c,h} - Z_{j,p,c,h}|.$$

By this changes, inter-cell material handling cost is calculated according to the movement between the cells.
• Modifications which are done by Tavakkoli et al. [13] are reconsidered in Safaei et al. [14], Mehrabad and Safaei [16], Bajestani et al. [17]. We used this method the formulation of our proposed model.

6. A genetic algorithm for solving DCF model

In this section, a genetic algorithm for solving DCF model is introduced. For designing the GA, six major factors are considered as follows:

6.1. Solution encoding (chromosome structure)

A chromosome or feasible solution for DCF problem consists of four genes as follow:

$$\left[ \begin{array}{c|c|c} X & Y & N \end{array} \right].$$

1. The gene related to assignment of part operation to machine is named matrix $[X]$. $X$ consists of $P$ matrices as $[X]_{H \times Op(i)}$ where $P$ is the number of products; $H$ is the number of periods and $Op(i)$ is the number of operations of part $i$.

$$[X] = \left[ X_{H \times Op(1)} \cdots X_{H \times Op(i)} \cdots X_{H \times Op(P)} \right].$$
A typical matrix as \([X]_{H \times Op(i)}\) in this arrangement is as follows:

\[
\begin{bmatrix}
X_{1,1}^{(0)} & X_{1,2}^{(0)} & \cdots & X_{1,j}^{(0)} & \cdots & X_{1,Op(i)}^{(0)} \\
X_{2,1}^{(0)} & X_{2,2}^{(0)} & \cdots & X_{2,j}^{(0)} & \cdots & X_{2,Op(i)}^{(0)} \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\
X_{H,1}^{(0)} & X_{H,2}^{(0)} & \cdots & X_{H,j}^{(0)} & \cdots & X_{H,Op(i)}^{(0)}
\end{bmatrix}
\]

In this matrix, for example, \(x_{H,j}^{(0)} = m\) means that operation \(j\) of part \(i\) during period \(H\) is done with machine type \(m\) \((m: 1, 2, \ldots, M)\).

2. The gene related to the assignment of part operation to cells is named matrix \([Y]\). \(Y\) consists of \(P\) matrices as \([Y]_{H \times Op(i)}\) where \(P\) is the number of products; \(H\) is the number of periods and \(Op(i)\) is the number of operations of part \(i\).

\[
[Y] : \begin{bmatrix}
[Y]_{H \times Op(1)} \cdots [Y]_{H \times Op(i)} \cdots [Y]_{H \times Op(p)}
\end{bmatrix}
\]

A typical matrix as \([Y]_{H \times Op(i)}\) in this arrangement is as follows:

\[
\begin{bmatrix}
Y_{1,1}^{(0)} & Y_{1,2}^{(0)} & \cdots & Y_{1,j}^{(0)} & \cdots & Y_{1,Op(i)}^{(0)} \\
Y_{2,1}^{(0)} & Y_{2,2}^{(0)} & \cdots & Y_{2,j}^{(0)} & \cdots & Y_{2,Op(i)}^{(0)} \\
\vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\
Y_{H,1}^{(0)} & Y_{H,2}^{(0)} & \cdots & Y_{H,j}^{(0)} & \cdots & Y_{H,Op(i)}^{(0)}
\end{bmatrix}
\]

In this matrix, for example, \(y_{H,j}^{(0)} = c\) means that operation \(j\) of part \(i\) during period \(H\) is done in cell number \(c\) and \(c\) \((c: 1, 2, \ldots, C)\).

3. The gene related to the number of machines being available in each cell is named matrix \([N]\). \(N\) consists of \(M\) matrices as \([N]_{H \times C}\) where \(C\) is the number of cells and \(H\) is the number of periods.

\[
[N] : \begin{bmatrix}
[N]_{H \times C}^{1} \cdots [N]_{H \times C}^{p} \cdots [N]_{H \times C}^{M}
\end{bmatrix}
\]

A typical matrix as \([N]_{H \times C}^{j}\) in this arrangement is as follows:

\[
\begin{bmatrix}
N_{1,1}^{(j)} & N_{1,2}^{(j)} & \cdots & N_{1,c}^{(j)} \\
N_{2,1}^{(j)} & N_{2,2}^{(j)} & \cdots & N_{2,c}^{(j)} \\
\vdots & \vdots & \ddots & \vdots \\
N_{H,1}^{(j)} & N_{H,2}^{(j)} & \cdots & N_{H,c}^{(j)}
\end{bmatrix}
\]

In this matrix, for example, \(N_{H,c}^{(j)} = a\) means that number of machine type \(m\) during period \(H\) in cell \(c\) is equal to \(a\).

4. The gene related to the number of machines being moved in each cell or the number of machines being moved out, is named matrix \([K]\). \(K\) consists of \(M\) matrices as \([K]_{H \times C}^{j}\) where \(C\) is the number of cells and \(H\) is the number of periods.

\[
[K] : \begin{bmatrix}
[K]_{H \times C}^{1} \cdots [K]_{H \times C}^{p} \cdots [K]_{H \times C}^{M}
\end{bmatrix}
\]

A typical matrix as \([K]_{H \times C}^{j}\) in this arrangement is as follows:

\[
\begin{bmatrix}
K_{1,1}^{(j)} & K_{1,2}^{(j)} & \cdots & K_{1,c}^{(j)} \\
K_{2,1}^{(j)} & K_{2,2}^{(j)} & \cdots & K_{2,c}^{(j)} \\
\vdots & \vdots & \ddots & \vdots \\
K_{H,1}^{(j)} & K_{H,2}^{(j)} & \cdots & K_{H,c}^{(j)}
\end{bmatrix}
\]

In this matrix, for example, \(K_{H,c}^{(j)} = a\), \(a > 0\) means that number of machine type \(m\) being moved in cell \(c\) during period \(H\) is equal to \(a\) and \(K_{H,c}^{(j)} = a\), \(a < 0\) means that number of machine type \(m\) being moved out from cell \(c\) during period \(H\) is equal to \(a\).
6.2. Initial population

A sequential strategy is used for obtaining the initial population. In the first step, part X of the first chromosome is generated randomly considering possibility of doing jobs on machines. After that, part Y of the chromosome is filled randomly too. Then the number of required machines is calculated regarding the generated pattern and parts N, K is filled consequently. Generated chromosomes might be infeasible according to the number of machines which are assigned to the cells because it is constrained by upper and lower bounds in mathematical formulation. Therefore a modification is needed to convert this infeasible solution to a feasible one. This job is done by assigning operations in most populated cells to the least populated ones.

6.3. Fitness value

The fitness value is criterion for the quality of the chromosome or feasible solution. In the proposed model, the fitness value is the same as the objective function of the model.

6.4. Selection strategies

In the proposed GA, three selection strategies are used. In the first strategy, some of the best chromosomes among parents are directly transmitted to the next generation according to their fitness values. This strategy is elitism which is used to let the best parents being alive for the next generation.

For crossover operator a mating pool is first generated and parent are selected from mating pool. And finally parents are selected for mutation randomly.

For operating crossover, it is required to select the most promising parents because better parent will have better offspring. Thus a normalization method is used to generate mating pool. For each generation mean and standard deviation of the objective function are calculated. Then the chromosomes which have less than or equal to mean value are transmitted to mating pool. This ensures that the best parents make the offsprings.

6.5. Improved GA operators

In this paper, the chromosome structure is formed as a matrix. Thus, the GA linear operators cannot be used to a matrix type as the traditional forms. These operators should be improved proportional to the matrix type. Therefore considering the nature of matrix, each of GA operators is considering as two named columnar, and districted. The cases of columnar and districted are described as follows:

1. Exercising of operation as columnar (in horizontal direct). In this case, two numbers are first selected randomly in row limits of relevant matrix. Then, the operation is exercised over obtained columns. For example in Fig. 2, rows 2 and 3 from matrix \( Y \) of elected chromosome are selected randomly and then the mutation operator is used.

2. Exercising of operation as block In this case, two numbers in the relevant matrix columnar limits and also two numbers in the relevant matrix row limits are selected randomly. Then, the operation is exercised over obtained district. For example in Fig. 3, the block obtained from cross of columns 2–4 and rows 3 and 4 related to two elected chromosomes is selected randomly and then the crossover operator is used.

6.6. Crossover operators

Crossover operators are as follows:

1. Districted crossover on gene X. In this case, a block of gene Y is first selected. Then every element of this block of one parent is replaced by other parent, considering possibility of doing jobs on machines.

![Fig. 2. columnar mutation based on working possibility.](image-url)
2. Districted crossover on gene \( Y \). In this case, a block of gene \( Y \) is first selected. Then every element of this block from one parent is replaced by the other parent.

3. Simultaneous districted crossover on gene \( X \) and \( Y \).

   In this case, two aforementioned crossovers are exercised on parents simultaneously.

6.7. Mutation operators

Mutation operators are as follows:

1. Districted mutation on gene \( X \). In this case, first a block of gene \( X \) is selected. Then one random machine type is selected for every element of this block, considering possibility of doing jobs on machines. Then the previous machine types are replaced by newly selected ones.

2. Districted mutation on gene \( Y \). In this case, a block of gene \( Y \) is first selected. Then every element of this block is filled randomly.

3. Simultaneous districted mutation on gene \( X \) and \( Y \). In this case, two aforementioned mutations are exercised on parent simultaneously.

6.8. Modification operation

Exercising every GA operator on part A, machine pattern is changed and sometimes leads to the condition that the number of machines existing in some cells exceeds upper and lower limits, so a modification operator is needed to convert this infeasible solution to a feasible one. This job is done by assigning operations in most populated cells to the least populated ones.

6.9. Stopping criteria

1. Number of generations. In this case, the algorithm terminates if the number of the generations exceeds a predefined number.

2. Time interval. In this case, the algorithm terminates if the current time of the algorithm exceeds a predefined solving time upper bound.

3. Improvement of the fitness value. In this case, the algorithm terminates if the improvement of the fitness value along a predefined number of the last generations being less than a given percentage.

6.10. GA algorithm

Step 1: Initialize parameters \( K \) (the number of chromosomes in each generation), \( G \) (the number of generation), \( e \) (percentage of the elitism) and \( \theta \) (the percentage of the crossover operation).

Step 2: Generate the initial population.

Step 3: Initialize counters \( r = 1 \) (population counter), \( g = 1 \) (generation counter).

Step 4: Calculate the number of chromosomes which should be transferred directly to the next generation:

\[
N_{elitism} = \lfloor (K - 1)e \rfloor + 1.
\]
Step 5: Calculate the number of times that each operation (mutation and crossover) should be used

\[
N_{crossover} = [(K - N_{elitism})\theta] + 1,
\]

\[
N_{mutation} = K - N_{elitism} - N_{crossover}.
\]

Step 6: Calculate the fitness values of the current population as \(F(X^h_1), F(X^h_2), \ldots, F(X^h_K)\).

Step 7: Normalize population fitness as \(Z_1, Z_2, \ldots, Z_k\), where \(Z_i = \frac{X^h_i - \mu}{\sigma} \) (\(\mu\) is the mean of the fitness value of the population and \(\sigma\) is the standard deviation of them).

Step 8: Transfer the \(N_{elitism}\) best solutions of the previous generation directly to the current generation and set \(r = N_{elitism} + 1\).

Step 9: Choose mating pool (solutions \(X_i\) in which \(Z_i < 0\)).

Step 10: Select two chromosomes of the mating pool randomly.

Step 11: Operate crossover on related parts of the selected chromosomes.

Step 12: Modify offspring till reaching to feasible solutions.

Step 13: If summation of offspring’s fitness values being less than that of parent’s, both of them are transferred to next generation and \(r = r + 2\) else if fitness value of one of offsprings being less than of both parents, just this offspring transferred to next generation and \(r = r + 1\).

Step 14: if \(r < N_{crossover} + N_{elitism}\) then go to Step 10.

Step 15: Select one of the chromosomes of the previous population.

Step 16: Operate mutation on related parts of the selected chromosome.

Step 17: Modify offspring till reaching to a feasible solution.

Step 18: Transfer chromosome directly to the next generation.

Step 19: if \(r < K\) then Set \(r = r + 1\) and go to Step 16.

Step 20: If \(g < \) number of generations or stopping criteria is met then Set \(g = g + 1\) and go to Step 6.

Step 21: Stop algorithm and report the best solution.

7. Computational result

A program based on the proposed GA is developed using Visual Basic.NET 2005 and all of these test problems are solved using this program. All of the computations are done on a PC Pentium IV 3 GHz AMD and 512 MB RAM DDR under win XP SP2. To demonstrate the effectiveness of proposed GA to solve improved DCF problem, firstly five test problems are designed which are solved using LINGO and optimal solution is found for them. The results of solving these problems using LINGO and proposed GA are shown and compared in Table 2.

As is shown in this table, Objective difference between the Proposed GA and Global Optimum, in the worst case is around 0.5% of global optimum value. This result shows that the proposed GA is able to find and report the near-optimal and promising solutions in a reasonable computational time.

### Table 2
Comparison of proposed GA and LINGO for solvable examples by LINGO.

<table>
<thead>
<tr>
<th>Problem (part, machine, cell, period)</th>
<th>LINGO 9.0</th>
<th>Proposed GA</th>
<th>Objective difference between the proposed GA and global optimum (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Var, Cons</td>
<td>Time</td>
<td>Objective</td>
</tr>
<tr>
<td>(2,2,2,2)</td>
<td>155, 114</td>
<td>0:0:1</td>
<td>11,520</td>
</tr>
<tr>
<td>(3,3,2,2)</td>
<td>266, 165</td>
<td>0:0:3:11</td>
<td>30,020</td>
</tr>
<tr>
<td>(3,3,3,3)</td>
<td>584, 342</td>
<td>0:0:35:18</td>
<td>22,930</td>
</tr>
<tr>
<td>(4,4,3,3)</td>
<td>885, 448</td>
<td>0:11:35:12</td>
<td>26,430</td>
</tr>
<tr>
<td>(5,5,3,3)</td>
<td>835, 292</td>
<td>0:10:41:20</td>
<td>53,197</td>
</tr>
</tbody>
</table>

### Table 3
Comparison of Proposed GA with LINGO for unsolvable examples by LINGO.

<table>
<thead>
<tr>
<th>Problem (Part, Machine, Cell, Period)</th>
<th>LINGO 9.0</th>
<th>Objective bound after half an hour</th>
<th>Proposed GA</th>
<th>Objective</th>
<th>Time</th>
<th>Objective difference between the proposed GA and objective bound (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>6,6,3,3</td>
<td>1649, 654</td>
<td>54,028</td>
<td>54,250</td>
<td>0:0:3:47</td>
<td>0.4</td>
<td></td>
</tr>
<tr>
<td>7,7,3,3</td>
<td>2112, 757</td>
<td>58,310</td>
<td>58,720</td>
<td>0:0:7:09</td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>8,8,3,3</td>
<td>1757, 582</td>
<td>42,625</td>
<td>43,085</td>
<td>0:0:4:32</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>9,9,3,3</td>
<td>3200, 972</td>
<td>55716.3</td>
<td>57,195</td>
<td>0:8:58</td>
<td>2.6</td>
<td></td>
</tr>
<tr>
<td>10,10,3,3</td>
<td>2565, 766</td>
<td>64,435</td>
<td>65,335</td>
<td>0:0:6:35</td>
<td>1.3</td>
<td></td>
</tr>
</tbody>
</table>
Increasing the dimension of the problem, solving them using LINGO is harder and it is derived from NP-Hardness of CF problem. To show the performance of proposed GA in solving this class of problems, five test problems which LINGO is unable to find global optimum for them are designed and solved using proposed GA. To validate the results obtained via GA, they are compared with Objective Bound found by LINGO after half an hour. These results are shown in Table 3.

As is shown in this table even for unsolvable problems by LINGO, the difference between the objective of Proposed GA and Objective Bound in the worst case is around 2.6% of Objective Bound value. It verifies that the proposed GA is able to find good results, which are dramatically near optimal. Moreover, computing time for these problems is less than 9 min and it can be concluded that proposed GA is efficient from the computational point of view.

8. Conclusion and future researches

In this paper, solving a CF problem in dynamic condition is discussed, shortcomings of previous models in the literature problem are removed and a new improved model is presented. As DCF model is NP-Hard, a new GA is developed to solve it. To validate this genetic algorithm, some test problems are designed and solved using LINGO and proposed GA. The comparison of the results showed that proposed GA is able to find near optimal solutions for problems in a more reasonable time than LINGO. To verify the applicability of this algorithm to solve larger problems which are not solvable using LINGO, some larger test problems are designed and solved using proposed GA. Obtained results showed that proposed GA is fast and finds near optimal solution in comparison with solution bounds found by LINGO after half an hour.

The proposed model is subjected to many assumptions which can be removed in future researches to increase its applicability. Moreover, it seems that in spite of good performance of proposed GA, designing a faster algorithm to solve proposed DCF model can be another area of work in future researches.

References