Genetic algorithm approach for solving a cell formation problem in cellular manufacturing

Iraj Mahdavi a,*, Mohammad Mahdi Paydar a, Maghsud Solimanpur b, Armaghan Heidarzade a,c

aDepartment of Industrial Engineering, Mazandaran University of Science and Technology, Tabarsi Street, Babol 47166-95635, Iran
bFaculty of Engineering, Urmia University, Urmia, Iran
cDepartment of Industrial Engineering, Payame Noor University, Sari, Iran

ARTICLE INFO

Keywords:
Cell formation
Cellular manufacturing
Mathematical model
Genetic algorithm
Group efficacy

ABSTRACT

Cellular manufacturing (CM) is an industrial application of group technology concept. One of the problems encountered in the implementation of CM is the cell formation problem (CFP). The CFP attempted here is to group machines and parts in dedicated manufacturing cells so that the number of voids and exceptional elements in cells are minimized. The proposed model, with nonlinear terms and integer variables, cannot be solved for real-sized problems efficiently due to its NP-hardness. To solve the model for real-sized applications, a genetic algorithm is proposed. Numerical examples show that the proposed method is efficient and effective in searching for optimal solutions. The results also indicate that the proposed approach performs well in terms of group efficacy compared to the well-known existing cell formation methods.

1. Introduction

Cellular manufacturing (CM) is a production system in which similar parts are classified into part families and dissimilar machines are assigned into machine cells 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. (Heragu, 1994; Weimerlov & Hyer, 1989). The cell formation problem (CFP) is one of the key issues in the design of CM (Soleymanpour, Vrat, & Shanker, 2002). In the past several years, many solution methods have been developed for solving CFP. A detailed review of the current researches and solution approaches for cell formation problem can be found in Joines, King, and Culbret (1996), Selim, Askin, and Vakharia (1998) and Singh (1993).

Most of the researches existing in the literature consider production flow analysis (PFA) data and attempt to group parts and machines in such a way that some performance criteria such as grouping efficiency, grouping efficacy, group technology efficiency, etc. are optimized (Soleymanpour et al., 2002). In PFA methods, mostly a part-machine incidence matrix, say \( A = [a_{ij}] \), is considered. If \( a_{ij} = 1 \), it means that part \( i \) requires processing on machine \( j \), otherwise \( a_{ij} = 0 \). The CFP is aimed to cluster all ‘ones’ in a block diagonal form. Since a binary representation of CFP has been attempted in this paper, some recent researches addressing the CFP as a binary part-machine incidence matrix are reviewed in the following.

Chen and Cheng (1995) considered a neural network based CFP in CM. They used an adaptive resonance theory (ART) based neural network to the cell formation problem in cellular manufacturing. The advantages of using an ART network over other conventional methods are its fast computation and the outstanding ability to handle large scale industrial problems. Mahdavi, Kaushal, and Chandra (2001) proposed a graph-neural network manufacturing approach for cell formation problems. Effort has been made to develop an algorithm that is more reliable than conventional methods. Their research has the ability to handle large scale industrial problems without the assumption of any parameter and the least exceptional elements in the presence of bottleneck machines and/or bottleneck parts. Soleymanpour et al. (2002) applied a transiently chaotic neural network approach (TCNN) to the design of CM. The TCNN algorithm aims at grouping similar parts and dissimilar machines to minimize the total number of exceptional elements and voids. Wang (2003) presented a linear assignment algorithm for machine-cell and part-family formation for the design of cellular manufacturing systems. The present approach begins with the determination of part-family or machine-cell representatives by means of comparing similarity coefficients between parts or machines and finding a set of the least similar parts or machines. Using the group representatives and associated...
similarity coefficients, a linear assignment model is formulated for solving the CFP by allocating the remaining parts or machines and maximizing a similarity index. Based on the formulated linear assignment model, a group formation algorithm was developed. Mahdavi, Javadi, Alipour, and Slomp (2007) proposed a new mathematical model for cell formation in cellular manufacturing system (CMS) based on cell utilization concept. The objective of the model is to minimize the number of voids in cells to achieve the higher performance of cell utilization. To validate and verify the proposed model, it is solved by the LINGO software using branch-and-bound (B&B). Wu, Chang, and Chung (2008) proposed a simple effective simulated annealing-based approach for obtaining machine-part groupings when the manufacturing system is represented by a 0–1 machine-part incidence matrix.

In this paper, we propose a mathematical programming model for cell formation in CMS. The objective of the model is to minimize the number of voids and exceptional elements (EEs) in cells to achieve the higher performance of cell utilization. However, mathematical models of this type may impose computational difficulties and may not be solvable using commercial optimization software for medium-to-large sized problems. Thus, efficient heuristic methods are required to solve the proposed model for problems of larger sizes. Here, we develop a heuristic method based on genetic algorithm to solve the proposed model.

The remainder of this paper is organized as follows. Section 2 provides a review of genetic algorithm (GA) approach in the CFP problem. In Section 3 the problem formulation of the proposed model is presented. Some preliminary computational work on small-scale examples is followed in Section 4. We elaborate the genetic algorithm (GA) approach for CMS in Section 5. The computational results are presented in Section 6 to illustrate the model and computational efficiency of the algorithm. The results show the superiority of the proposed algorithm over the well-known methods. Conclusions are given in Section 7.

2. Genetic algorithm

GA is one of the new optimization approaches that attempt to mimic natural processes in order to create general purpose optimization procedures (Goldberg, 1989). GA, developed by Holland (1975), have been used extensively as an alternative method for solving different optimization problems in a wide variety of application domains including engineering, biology, economics, agriculture, business, telecommunications, and manufacturing (Gen & Cheng, 1997; Goldberg, 1989; Man, Tang, & Kwong, 1999). As a general-purpose search method, a GA combines elements of directed and stochastic search for exploring and exploiting the search space to obtain good solutions. In contrast to other stochastic searches, genetic algorithms (GAs) have the following unique features: implicit parallelism, population-based search, independence of gradient information, and flexibility to hybridize with domain-dependent heuristics. These features often make GAs a preferable choice over traditional heuristics.

GAs have been effectively used to solve cell formation problems of CM. Venugopal and Narendran (1992) modeled the cell formation problem based on minimizing the total cell load variation using GAs. Joines, Culbret, and King (1996) developed an integer program for cell design problem and used GA to solve the formulated model. Zhao and Wu (2000) present a GA for cell formation with multiple routes and objectives. Onwubolu and Mutungi (2001) develop a GA approach taking into account cell-load variation. Brown and Sumichrast (2001) propose an approach using grouping GA. Uddin and Shanker (2002) address a generalized grouping problem, where each part has more than one process route. The problem is formulated as an integer programming problem and a procedure based on a GA is suggested as a solution methodology. Goncalves and Resende (2004) presented a new approach for obtaining machine cells and product families. The approach combines a local search heuristic with a GA.

3. Problem formulation

In this section, we formulate a new nonlinear mathematical model based on machine-part incidence matrix. The proposed model deals with the minimization of EEs and number of voids in cells to achieve the higher performance of cell utilization. EEs are treated as those ‘ones’ falling outside the block-diagonal cluster of ‘ones’. Voids are treated as those ‘zeros’ falling within the block-diagonal cluster of ‘ones’. The cell utilization concept is considered as the number of non-zero elements of block-diagonal divided by the block-diagonal matrix size of each cell.

3.1. Indexing sets

\[ i: \text{index for parts } i = 1, ..., P \]
\[ j: \text{index for machines } j = 1, ..., M \]
\[ k: \text{index for cells } k = 1, ..., C \]

3.2. Parameters

\[ r_{ij} = \begin{cases} 1 & \text{if part } i \text{ is to be processed on machine } j, \\ 0 & \text{otherwise}. \end{cases} \]

3.3. Decision variables

\[ Y_{jk} = \begin{cases} 1 & \text{if machine } j \text{ is assigned to cell } k, \\ 0 & \text{otherwise}. \end{cases} \]
\[ Z_{ik} = \begin{cases} 1 & \text{if part } i \text{ is assigned to cell } k, \\ 0 & \text{otherwise}. \end{cases} \]

3.4. Objective function

We describe the objective function as

\[ \min Z = \sum_{k=1}^{C} \left( \sum_{i=1}^{P} \sum_{j=1}^{M} Z_{ik} Y_{jk} - \sum_{i=1}^{P} \sum_{j=1}^{M} Z_{ik} Y_{jk} r_{ij} \right) + \sum_{i=1}^{P} \left( \sum_{j=1}^{M} r_{ij} - \sum_{k=1}^{C} Z_{ik} Y_{jk} r_{ij} \right) \] (1)

3.5. Constraints

\[ \sum_{k=1}^{C} Y_{jk} = 1 \quad \forall j \] (2)
\[ \sum_{i=1}^{P} Z_{ik} = 1 \quad \forall i \] (3)
\[ \sum_{j=1}^{M} Z_{ik} Y_{jk} r_{ij} \geq \min_{T_{ik}} \times \sum_{j=1}^{M} Z_{ik} Y_{jk} \quad \forall k \] (4)
\[ Z_{ik}, Y_{jk} \in \{0, 1\} \quad \forall i, j, k \] (5)

The objective function given in Eq. (1) is to minimize the total number of voids and EEs. The first term computes the total number of voids and the second term computes the total number of EEs. More-
over, the term $\sum_{i=1}^{p} \sum_{j=1}^{m} r_{ij}$ from the objective function is a constant value and indicates the total number of ‘ones’ in the machine-part incidence matrix. So, the objective function could be simplified as follows:

$$\text{Min } Z = \sum_{i=1}^{p} \sum_{j=1}^{m} (1 - 2r_{ij}) Z_{ik} Y_{jk}$$

Eq. (2) guarantees that each machine must be assigned to one cell only. Eq. (3) guarantees that each part must be assigned to one cell only. Constraint (4) guarantees that each cell achieves a minimum utilization of interest.

4. Global optimization experience

Four problems were solved in order to validate the proposed mathematical formulation that represents the CM model of this research. The models were solved using LINGO 8 on a PC Pentium IV 2.1 GHz with 512 Mb of RAM, with the results shown in Table 1. It must be noted that the nonlinear mathematical model proposed in this paper can be linearized (Mahdavi et al., 2007). By linearization of the mathematical model, LINGO finds the global optimum solution of the small-sized problems in a short time. However, some of the problems cannot be solved in reasonable time using Lingo, as shown in Table 1, because of excessive number of variables and constraints.

For the first three scenarios, global optimum solutions were obtained after the shown computational time. No solution was obtained for scenario four, even after twenty hours (36,014,381 iterations). This fact reveals that a heuristic approach is needed to optimally solve the proposed model. In Section 5, a GA is developed to solve the proposed model.

5. The GA approach

GA is based on an analogy to the phenomenon of natural selection in biology. First, a chromosome structure is defined to represent the solutions of the problem. GA’s can be implemented in a variety of ways. The excellent books by Davis (1991) and Goldberg (1989) describe many possible variants of GAs. Using this structure, an initial solution population is generated, either randomly or using a given heuristic. Then, members of the population are selected, based on an evaluation function, called fitness that, associates a value to each member according to its objective function. The higher a member’s fitness value, the more likely it is to be selected. Thus, the less fit individuals are replaced by those who perform better. Genetic operators are then applied to the selected members to produce a new population generation. This process is repeated until a certain number of iterations are reached.

Fig. 1 shows the general procedure of GAs.

The main components of GA for implementation identify six components:

1. The scheme for coding.
2. The initial population.
3. An adaptation function for evaluating the fitness of each member of the population.
4. Selection procedure.
5. The genetic operators used for producing a new generation.
6. Certain control parameter values (e.g. population size, number of iterations, genetic operator probabilities).

5.1. The scheme for coding

For any GA implementation, the first stage is to map solution characteristics in the format of a chromosome string. Each

<table>
<thead>
<tr>
<th>Article/attributes</th>
<th>No. of part types</th>
<th>No. of machine types</th>
<th>No. of cells</th>
<th>No. of variables</th>
<th>No. of constraints</th>
<th>Computation time (s)</th>
<th>Solution status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chen and Cheng (1995)</td>
<td>6</td>
<td>6</td>
<td>2</td>
<td>114</td>
<td>161</td>
<td>1</td>
<td>Global optimum</td>
</tr>
<tr>
<td>Chen and Cheng (1995)</td>
<td>10</td>
<td>10</td>
<td>3</td>
<td>400</td>
<td>627</td>
<td>35</td>
<td>Global optimum</td>
</tr>
<tr>
<td>Chandrasekharan and Rajagopalan (1986a)</td>
<td>8</td>
<td>20</td>
<td>3</td>
<td>644</td>
<td>995</td>
<td>37</td>
<td>Global optimum</td>
</tr>
<tr>
<td>Standel (1985)</td>
<td>14</td>
<td>24</td>
<td>6</td>
<td>2412</td>
<td>4083</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Table 1: The results of the linearized model solved by LINGO 8

Fig. 1. A generic framework for GAs.
chromosome is made up of a sequence of genes from a certain alphabet. The alphabet can be a set of binary numbers, real numbers, integers, symbols, or matrices (Goldberg, 1989). The representation scheme determines not only how effective the problem is structured, but also how efficient the genetic operators can be used.

In this paper, a direct coding scheme is used, that is the allele of each gene represents the cell number to which the machine or part belongs (Chu & Tsai, 2001; Venugopal & Narendran, 1992). The chromosome representation consists of two sections: the first section represents the parts and the second stands for machines. The chromosome used for the CF problem can be represented as shown in bellow where \( P_i \) denotes the group to which part \( i \) is assigned and \( M_j \) denotes the group to which machine \( j \) is assigned has been assigned.

\[
\begin{align*}
P_1 & \quad P_2 & \quad P_3 & \quad \ldots & \quad P_r & \quad M_1 & \quad M_2 & \quad M_3 & \quad \ldots & \quad M_M
\end{align*}
\]

For example, consider the following chromosome for a cell formation problem with 10 parts and 10 machines:

2 2 3 3 1 1 1 2 2 2 1 1 2 3 3 1 1 1

This chromosome indicates three cells that part 1 belongs to cell 2, machine 3 belongs to cell 1, and so on. As such, cell 1 contains parts \([6, 7, 8]\) and machines \([3, 4, 8, 9, 10]\), cell 2 contains parts \([1, 2, 3, 9, 10]\) and machines \([1, 2, 5]\), and cell 3 contains parts \([4, 5]\) and machines \([3, 7, 8]\). Note that the part and machine portions of the chromosome are fixed in length based on the size of the problem.

5.2. The initial population

The second stage of GA implementation is to generate a set of initial solutions, called population. The number of initial solutions to be included in the population is called population size. The initial population is generated only once at the beginning for the first generation of the GA. Determining the proper population size is a major decision in GA implementation. If the selected number is too small, we may not be able to obtain a good solution. On the other hand, if the number is too large, it may consume too much CPU time to obtain a better solution (Back, Fogel, & Michalawecz, 1997). A special procedure was developed in this research to generate a random initial population while constraints 2 and 3 are satisfied, i.e. each machine and each part should be assigned to only one cell.

5.3. Fitness function

In GA implementation, a fitness function is used to evaluate and reproduce new chromosomes, called offspring for the generations to come. The purpose of the fitness function is to measure the goodness of the candidate solutions in the population with respect to the objective and constraint functions of the model. The fitness value of a chromosome in the proposed algorithm is the total number of voids and EEs in all cells.

5.4. Selection rule

The Roulette Wheel selection procedure, as proposed by Goldberg (1989), is the selection strategy used in the proposed algorithm. The goal of selection strategy is to allow the “fittest” individuals to be considered more often to reproduce children for the next generation. Each individual is assigned a probability of being selected based on its fitness value. Although better individuals will have a higher selection probability, all individuals in the population will have a chance to be selected. Hence, we selected both parents randomly after ranking individuals based on their fitness and giving weighted probability of selection to each individual in the population.

5.5. Genetic operators

Reproduction is carried out by using crossover and mutation operators on the selected parents to produce new offspring. We developed crossover and mutation operators for the proposed algorithm as discussed in the following.

5.5.1. Crossover operator

Crossover combines information from two parents such that the two children have a “resemblance” to each parent. Traditionally, standard crossovers such as one-point, two-point, and uniform are used in GA models (Goldberg, 1989). If application of these operators results in illegal chromosomes (for instance, for order permutation problems), variations such as partially matched crossover (PMX), order crossover (OX), cycle crossover (CX), and many others may be used if applicable (Gen & Cheng, 1997). The crossover operator should be capable of producing a new feasible solution (i.e., a child) by combining good characteristics of both parents. Preferably, the child should be considerably different from each parent. We applied two crossover operators, called the (1) simple crossover, (2) uniform crossover (Ji et al., 2006). These operators are discussed in detail in the following.

5.5.1.1. Simple crossover. Two individuals, called parents, are randomly selected from the population. Then a number between 1 and \( M + P \) (\( M \) is the number of machines and \( P \) is the number of parts) is selected. Suppose this number is \( r \). Then, the genes \( r \) to \( M + P \) of both parents are exchanged to build two new chromosomes (children). Fig. 2 shows the simple crossover operation.

5.5.1.2. Uniform crossover. For every pair of randomly selected parents, a small proportion of randomly selected genes are exchanged. The crossover process is illustrated in Fig. 3. Individuals A and B produce C and D after applying the crossover. We define A as the direct parent of C, and B as the direct parent of D.

5.5.2. Mutation operator

The traditional mutation operator mutates the gene’s value randomly according to a small probability of mutation; thus, it is merely a random walk and does not guarantee a positive direction toward the optimal solution. The proposed heuristic mutation remedies this deficiency. In this scheme an individual is randomly chosen from the population. Then a random number like \( r \) between 1 and \( M + P \) (\( M \) is the number of machines and \( P \) is the number of parts) is selected. The pseudo code of the proposed mutation operator is as follows.

For example, consider the following chromosome for a cell formation problem with 10 parts and 10 machines:
If \( a_r = 1 \) (\( a_r \) is \( r \)th gene of the selected chromosome)
   Set \( a_r = C \) (\( C \) is the total number of cells)
Else
   If \( a_r = C \)
      \( a_r = 1 \)
   Else
      Generate a random number like \( r_x \) between 0 and 1.
      If \( r_x < 0.5 \)
         \( a_r = a_r - 1 \)
      Else
         \( a_r = a_r + 1 \)

5.6. Parameters

The parameters required to run the algorithm are population size, number of generations, number of iterations, crossover and mutation probabilities. These parameters have a crucial role in the performance of the GAs.

The number of generations is a function of the size of the problem at hand. As the solution space increases, the GA will require a higher number of generations to possibly reach a convergence point. Population size may vary depending on the application. The number of iterations must be set in such a way to allow the GA to complete the convergence process. The crossover operator has a significant effect on the performance of GA and hence a relatively large probability value is considered for this parameter. Mutation operator is basically used to maintain diversity in the population and a low probability is set to this parameter.

6. Computational results

To demonstrate the performance of the proposed algorithm, we tested the GA on 22 benchmark problems collected from the literature. The meta-heuristic algorithms were developed using MATLAB 7 and run on a PC Pentium IV, 2.1 GHz speed with 512 Mb of RAM. The sources and input parameters for these problems are shown in Table 2. To cover different sizes, problems with small size (e.g. 5 \times 7), medium size (e.g. 10 \times 15, 14 \times 24) and large size (e.g. 24 \times 40, 40 \times 100) have been selected. For simplicity, minimum utilization are the same for all cells in each problem as shown in Table 2.

A summary of all GA parameters used for solving the benchmark problems are given in Table 3. The population size increases when the problem size increases. The most commonly used objectives in cell formation are to minimize intercellular movements and maximize machines' utilization (Zolfaghari & Liang, 1997). The presence of EEs causes intercellular movements. On the contrary, forcing EEs to go to

<table>
<thead>
<tr>
<th>No.</th>
<th>Problem source</th>
<th>Source</th>
<th>No. of machines</th>
<th>No. of parts</th>
<th>Parameter</th>
<th>No. of cells</th>
<th>Min. Utk</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Fig. 4a in Waghodekar and Sahu (1984)</td>
<td>5</td>
<td>7</td>
<td>2</td>
<td>0.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Seifoddini (1989)</td>
<td>5</td>
<td>18</td>
<td>2</td>
<td>0.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Kusiak and Cho (1992)</td>
<td>6</td>
<td>8</td>
<td>2</td>
<td>0.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Kusiak and Chow (1987)</td>
<td>7</td>
<td>11</td>
<td>5</td>
<td>0.9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Boctor (1991)</td>
<td>7</td>
<td>11</td>
<td>4</td>
<td>0.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Chandrasekharan and Rajagopalan (1986a)</td>
<td>8</td>
<td>20</td>
<td>3</td>
<td>0.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Chandrasekharan and Rajagopalan (1986b)</td>
<td>8</td>
<td>20</td>
<td>2</td>
<td>0.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Mosier and Taube (1985)</td>
<td>10</td>
<td>10</td>
<td>5</td>
<td>0.75</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Chan and Milner (1982)</td>
<td>10</td>
<td>15</td>
<td>3</td>
<td>0.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Stanfel (1985)</td>
<td>14</td>
<td>24</td>
<td>7</td>
<td>0.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>King (1980)</td>
<td>16</td>
<td>43</td>
<td>7</td>
<td>0.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Mosier and Taube (1985)</td>
<td>20</td>
<td>20</td>
<td>5</td>
<td>0.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Carrie (1973)</td>
<td>20</td>
<td>35</td>
<td>5</td>
<td>0.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Data set 1 in Chandrasekharan and Rajagopalan (1989)</td>
<td>24</td>
<td>40</td>
<td>7</td>
<td>0.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Data set 2 in Chandrasekharan and Rajagopalan (1989)</td>
<td>24</td>
<td>40</td>
<td>7</td>
<td>0.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Data set 3 in Chandrasekharan and Rajagopalan (1989)</td>
<td>24</td>
<td>40</td>
<td>7</td>
<td>0.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Data set 5 in Chandrasekharan and Rajagopalan (1989)</td>
<td>24</td>
<td>40</td>
<td>10</td>
<td>0.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>Data set 6 in Chandrasekharan and Rajagopalan (1989)</td>
<td>24</td>
<td>40</td>
<td>12</td>
<td>0.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>Data set 7 in Chandrasekharan and Rajagopalan (1989)</td>
<td>24</td>
<td>40</td>
<td>12</td>
<td>0.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>Data set 8 in Stanfel (1985)</td>
<td>30</td>
<td>50</td>
<td>13</td>
<td>0.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Fig. 5 in Stanfel (1985)</td>
<td>30</td>
<td>50</td>
<td>14</td>
<td>0.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>Data set 9 in Stanfel (1985)</td>
<td>40</td>
<td>100</td>
<td>10</td>
<td>0.5</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Please cite this article in press as: Mahdavi, I., et al., Genetic algorithm approach for solving a cell formation problem ... Expert Systems with Applications (2008), doi:10.1016/j.eswa.2008.07.054
manufacturing cells reduces the utilization of machines. Therefore, a trade-off between these conflicting objectives is the major problem of interest in the design of manufacturing cells. In order to reduce intercellular movements, the number of 'ones' out of the diagonal blocks in the machine-part incidence matrix should be minimized. Simultaneously, to increase utilization of machines, the number of 'zeros' inside the diagonal blocks is to be minimized. Simultaneously concerns the intended objectives and it has been widely used in the literature.

Although grouping efficiency is used as a measure of the quality of solutions, it suffers some limitations. For example, a very bad solution with many EEs often shows efficiency figures around 75%. Also, for large matrices, the denominator of the first term is more or less of the same order. When the matrix size increases, the effect of EEs becomes smaller, and in some cases, the effect of inter-cell moves is not reflected in the grouping efficiency (Cheng, Gupta, Lee, & Wong, 1998; Goncalves & Resende, 2004).

Therefore, Kumar and Chandrasekharan (1990) proposed a group-efficiency measure. This measure simultaneously concerns the intended objectives and it has been widely used in the literature.

The results of the proposed model are compared with those of the following methods wherever the grouping efficacy obtained from the application of first five these methods to above problems are available in Goncalves and Resende (2004).

- ZODIAC method (Chandrasekharan & Rajagopalan, 1987),
- GRAFICS method (Srinivasan & Narendra, 1991),
- GATSP-Genetic algorithm (Cheng et al., 1998),
- GA-Genetic algorithm (Onwubolu & Mutingi, 2001),
- EA-evolutionary algorithm (Goncalves & Resende, 2004),
- SA-simulated annealing (Wu et al., 2008).

Table 4 shows the grouping efficacy of the solutions obtained by the methods mentioned above and the proposed algorithm. The solutions obtained by the GRAFICS method for problems 2, 3 and 5 were not available.

As seen in Table 4, in all the benchmark problems, the grouping efficacy of the solution obtained by the proposed method is either better than that of other methods or it is equal to the best one. In six problems, namely 10, 11, 17, 18, 19 and 21, the grouping efficacy of the solution obtained by the proposed method is better than that of all other methods. In other words, the proposed method outperforms all the other methods and the best solutions for these problems are reported in this paper for the first time. In eleven problems, namely 1, 2, 4, 5, 7, 8, 12, 13, 16, 20 and 22, the solution obtained by the proposed method is as good as the best solution available in the literature. In five problems, namely 3, 6, 9, 14 and 15, all the methods have obtained the same grouping efficacy.

7. Conclusions

In this paper a new mathematical model for the CFP based on cell utilization concept in CMS is introduced. The proposed model determines the cell configuration with the aim of minimizing the EEs and the number of voids in cells simultaneously. An efficient algorithm based on GA was designed to solve the mathematical model. In order to verify the performance of the GA approach, we solved 22 problems selected from the literature.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chromosome size</td>
<td>Number of machines and parts</td>
</tr>
<tr>
<td>Population size</td>
<td>50–200</td>
</tr>
<tr>
<td>Crossover probability</td>
<td>0.70–0.80</td>
</tr>
<tr>
<td>Mutation probability</td>
<td>0.01–0.1</td>
</tr>
<tr>
<td>Number of generations</td>
<td>Variable</td>
</tr>
</tbody>
</table>

Table 3
The values used for different GA parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ZODIAC</td>
<td>GRAFICS</td>
</tr>
<tr>
<td>GATSP</td>
<td>GA</td>
</tr>
<tr>
<td>EA</td>
<td>SA</td>
</tr>
</tbody>
</table>

Table 4
Performance of the proposed algorithm compared to other algorithm

<table>
<thead>
<tr>
<th>No.</th>
<th>Size</th>
<th>ZODIAC</th>
<th>GRAFICS</th>
<th>GATSP</th>
<th>GA</th>
<th>EA</th>
<th>SA</th>
<th>Max. solution of proposed approach</th>
<th>CPU time (s) of proposed approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5 x 7</td>
<td>56.52</td>
<td>60.87</td>
<td>68.00</td>
<td>62.50</td>
<td>52.60</td>
<td>69.57</td>
<td>69.57</td>
<td>0.006</td>
</tr>
<tr>
<td>2</td>
<td>5 x 18</td>
<td>77.36</td>
<td>–</td>
<td>77.36</td>
<td>77.36</td>
<td>79.59</td>
<td>79.59</td>
<td>79.59</td>
<td>0.018</td>
</tr>
<tr>
<td>3</td>
<td>6 x 8</td>
<td>76.92</td>
<td>–</td>
<td>76.92</td>
<td>76.92</td>
<td>76.92</td>
<td>76.92</td>
<td>76.92</td>
<td>0.009</td>
</tr>
<tr>
<td>4</td>
<td>7 x 11</td>
<td>39.14</td>
<td>53.12</td>
<td>46.88</td>
<td>50.00</td>
<td>53.13</td>
<td>60.87</td>
<td>60.87</td>
<td>0.015</td>
</tr>
<tr>
<td>5</td>
<td>7 x 11</td>
<td>70.37</td>
<td>–</td>
<td>70.37</td>
<td>70.37</td>
<td>70.37</td>
<td>70.83</td>
<td>70.83</td>
<td>0.021</td>
</tr>
<tr>
<td>6</td>
<td>8 x 20</td>
<td>85.24</td>
<td>85.24</td>
<td>85.24</td>
<td>85.24</td>
<td>85.24</td>
<td>85.25</td>
<td>85.25</td>
<td>0.028</td>
</tr>
<tr>
<td>7</td>
<td>8 x 20</td>
<td>58.33</td>
<td>58.13</td>
<td>58.33</td>
<td>55.91</td>
<td>58.72</td>
<td>58.72</td>
<td>58.72</td>
<td>0.032</td>
</tr>
<tr>
<td>8</td>
<td>10 x 10</td>
<td>70.59</td>
<td>70.59</td>
<td>70.59</td>
<td>72.79</td>
<td>70.59</td>
<td>70.59</td>
<td>70.59</td>
<td>0.025</td>
</tr>
<tr>
<td>9</td>
<td>10 x 15</td>
<td>92.00</td>
<td>92.00</td>
<td>92.00</td>
<td>92.00</td>
<td>92.00</td>
<td>92.00</td>
<td>92.00</td>
<td>0.026</td>
</tr>
<tr>
<td>10</td>
<td>14 x 24</td>
<td>65.55</td>
<td>65.55</td>
<td>67.44</td>
<td>63.48</td>
<td>69.33</td>
<td>71.21</td>
<td>71.21</td>
<td>0.571</td>
</tr>
<tr>
<td>11</td>
<td>16 x 43</td>
<td>53.76</td>
<td>54.39</td>
<td>53.98</td>
<td>86.25</td>
<td>58.46</td>
<td>52.44</td>
<td>56.13</td>
<td>1.532</td>
</tr>
<tr>
<td>12</td>
<td>20 x 20</td>
<td>21.63</td>
<td>38.26</td>
<td>37.12</td>
<td>34.16</td>
<td>42.94</td>
<td>41.04</td>
<td>41.04</td>
<td>0.620</td>
</tr>
<tr>
<td>13</td>
<td>20 x 35</td>
<td>75.14</td>
<td>75.14</td>
<td>75.28</td>
<td>66.30</td>
<td>76.22</td>
<td>78.40</td>
<td>78.40</td>
<td>1.252</td>
</tr>
<tr>
<td>14</td>
<td>24 x 40</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
<td>100.00</td>
<td>1.601</td>
</tr>
<tr>
<td>15</td>
<td>24 x 40</td>
<td>85.11</td>
<td>85.10</td>
<td>85.11</td>
<td>85.11</td>
<td>85.11</td>
<td>85.11</td>
<td>85.11</td>
<td>1.923</td>
</tr>
<tr>
<td>16</td>
<td>24 x 40</td>
<td>73.51</td>
<td>73.51</td>
<td>73.03</td>
<td>73.03</td>
<td>73.51</td>
<td>73.51</td>
<td>73.51</td>
<td>1.477</td>
</tr>
<tr>
<td>17</td>
<td>24 x 40</td>
<td>20.42</td>
<td>43.27</td>
<td>49.37</td>
<td>37.62</td>
<td>51.88</td>
<td>52.44</td>
<td>52.87</td>
<td>3.256</td>
</tr>
<tr>
<td>18</td>
<td>24 x 40</td>
<td>18.23</td>
<td>44.51</td>
<td>44.67</td>
<td>34.76</td>
<td>46.69</td>
<td>47.13</td>
<td>48.95</td>
<td>6.242</td>
</tr>
<tr>
<td>19</td>
<td>24 x 40</td>
<td>17.61</td>
<td>41.67</td>
<td>42.50</td>
<td>34.06</td>
<td>44.75</td>
<td>44.64</td>
<td>47.26</td>
<td>11.233</td>
</tr>
<tr>
<td>20</td>
<td>30 x 50</td>
<td>46.06</td>
<td>56.32</td>
<td>56.61</td>
<td>48.28</td>
<td>59.21</td>
<td>60.12</td>
<td>60.12</td>
<td>19.304</td>
</tr>
<tr>
<td>21</td>
<td>30 x 50</td>
<td>21.11</td>
<td>47.96</td>
<td>45.93</td>
<td>37.55</td>
<td>50.48</td>
<td>50.51</td>
<td>50.83</td>
<td>22.212</td>
</tr>
<tr>
<td>22</td>
<td>40 x 100</td>
<td>83.92</td>
<td>83.92</td>
<td>84.03</td>
<td>83.90</td>
<td>84.03</td>
<td>84.03</td>
<td>84.03</td>
<td>99.631</td>
</tr>
</tbody>
</table>

* Please cite this article in press as: Mahdavi, I. et al., Genetic algorithm approach for solving a cell formation problem ... Expert Systems with Applications (2008), doi:10.1016/j.eswa.2008.07.054
Computational experience showed the superiority of the proposed methodology in the grouping of parts and machines as compared with ZODIAC, GRAFICS, GATSP, GA, EA, SA methods.

References


