International Journal of Natural and Engineering Sciences 3 (1): 73-78, 2009
ISSN: 1307-1149, www.nobel.gen.tr

A grouping genetic algorithm for the cell formation problem

Mojtaba SALEHI[1] Reza Tavakkoli MOGHADDAM[2]

[1] Department of Industrial Engineering, University of Bojnourd, Bojnourd, Iran
[2] Department of Industrial Engineering, University of Tehran, Iran

Corresponding Author
e-mail: m_salehi61@yahoo.com

Received: August 07, 2008
Accepted: December 15, 2008

Abstract

The cell formation problem (CFP) consists of constructing a set of machine cells and their corresponding product families with the objective of minimizing the inter-cell movement of the products while maximizing machine utilization. This paper presents a grouping genetic algorithm for the cell formation problem that uses grouping efficiency as a measure. We solve the CFP without predetermination of the number of cells. We also make some effort to improve the efficiency of our algorithm with respect to initialization of the population, keeping crossover operator from cloning . Computational results using the grouping efficacy measure for a set of cell formation problems from the literature are presented. The algorithm developed performs well on all test problem, exceeding or matching the solution quality of the results presented in previous literature for most problems.

Key words: Grouping Technology, Cellular Manufacturing Systems, block diagonalization problem, Grouping Efficiency, genetic algorithm

INTRODUCTION

Group Technology (GT) can be defined as a disciplined approach to identifying items such as parts, processes, and machines by their attributes; analyzing those attributes by looking for similarities between and among items; grouping the items into families according to similarities; and, finally, increasing the efficiency and effectiveness of managing the items by taking advantage of the similarities [1]. Cellular manufacturing (CM) is one of the major applications of group technology. CM is described as a manufacturing procedure which produces part families within a single line or cell of machines serviced by operators and/or robots that function only within the line or cell. The foremost problem for cellular manufacturing system (CMS) design is cell formation (CF), which groups the machines into machine cells and parts into part families. The cell formation problem (CFP) started with Burbidge [2], who introduced the first method of production flow analysis (PFA). For decades, many methods for the CFP have been reported and can be classified into the following[3]:

- Array-based methods
- Clustering methods
- Mathematical programming-based methods
- Graph theoretic methods
- Neural network-based methods
- Search methods such as Tabu search, Simulated annealing, Genetic Algorithm (GA)

Since our proposed method is based on GA, a review is detailed in the next section. GA is one of the meta-heuristic algorithms, which was introduced by Holland [4]. Genetic algorithms (GAs) are based on evolutionary principles such as natural selection and survival of the fittest. GAs attempt to evolve a population of solutions by giving preference for survival to high-quality solutions, while allowing some lower quality solutions to survive in order to maintain a level of diversity in the population. This process provides good solutions without premature convergence to local optima [5]. GAs have been used in a wide range of practical fields including design, scheduling, system configuration, financial portfolio management, adaptive control systems and noisy data interpretation.

The CFP is an NP-hard problem, in which a given machine part incidence matrix is explored with the objective of creating cells in which the intra-cell machine utilization is maximized and the inter-cell movement of the parts is minimized. Although some of the optimization algorithms can find the optimal solution for small- and medium-sized problems, they will suffer from the disadvantage that the memory and computational time requirements are extremely high and increase exponentially as the problem size increases. On the other hand, the objective of CFP in the real world is multiple. GAs are capable of dealing with a multi-objective problem and searching large regions of the solution's space while being less susceptible to becoming trapped in local optima. Grouping genetic algorithms (GGA) are a modification of traditional GAs designed specifically for grouping problems[6].

The first GA-based solution for the CFP was developed by Venugopal and Narendran [7]. They modeled the CFP with minimization of the total cell load variation and intercell part flows as an objective function and applied GA encoding to the problem on a machine–cell string with a given total number of

manufacturing cells. Billo et al. [8] showed another encoding method that contains a string of part numbers, where cell locations are identified with a vertical slash (|) that designates the cutoff for the part numbers comprising the cell. Such an example appears as 231|546|978. Joines et al. [9] formulate the cell formation problem as an integer-programming problem and propose a GA as a solution methodology. Cheng et al. [10] and Mak et al. [11] formulate the problem as a travelling salesman problem and solve the model using a genetic algorithm. Rao et al. [12] encoded the machine–cell chromosome into 0–1 combinations, in which the predetermined number of cells is needed. Zhao and Wu [13] present a genetic algorithm for cell formation with multiple routes and objectives. Onwubolu and Mutingi [14] develop a genetic algorithm approach taking into account cell-load variation. Uddin and Shanker [15] address a generalized grouping problem, where each part has more than one process route.

Realizing the drawbacks of applying a traditional GA to grouping problems, Falkenauer [6] modified the encoding strategy, the crossover operator and the mutation operator of traditional GAs to handle the special structure of grouping problems. The grouping genetic algorithm (GGA) is popular because of its efficiency in combinational grouping problems such as the bin packing problem, the graph coloring problem, and also the CFP. The GGA differs from the classic GA in encoding scheme and genetic operators suitable for the chromosomes. De Lit et al. [16] first solved the CFP using the GGA. Eduardo Vila and et al. [17] proposed GGA algorithm that had several new features such as the chromosome codification scheme, correction mechanism and the crossover and mutation operators. Tabitha L. James and et al. [18] presented a hybrid grouping genetic algorithm (HGGA) that combines a local search with a GGA. Most previous researchers decided on the number of cells based on their experience or compared several sets of numbers of cells from 1 to the maximum number, which is equal to the total number of machines, and made a choice on the best result.

In this paper, we develop a grouping genetic algorithm (GGA) for the machine-part cell formation problem. Using the grouping efficacy measure of Kumar and Chandrasekharan [19], we demonstrate that this approach not only reduces the variability in solution quality from that of a traditional GGA, but also improves the solution quality over the GGA for many problems. On the other hand, proposed GGA can solve the CFP without predetermination of the number of cells. We test 14 problems from the CFP literature and provide comparisons against several algorithms from the literature. The GGA is shown to perform well against previously reported solution methodologies, exceeding or matching the solutions found for many of the test problems.

The remainder of the paper is organized as follows. In the Material and Methods section, first the concept of Grouping Efficacy as a performance measure for the CFP is described and a model for the cell formation problem is proposed. Finally in this section we explain the particular GGA step by step. The Results and Discussion section gives a numerical example and compares studies. Conclusions are given in final Section.

MATERIALS AND METHODS

The Grouping Efficacy concept: In this paper, we attempt to solve the CFP as a zero one block diagonalization problem (BDP), to minimize inter-cellular movement and maximize the utilization of the machines within a cell, where a 0–1 machine-part incidence matrix (see Fig. 1) is the usual input data to the BDP. In the matrix, $a_{i,j}$ =1 if machine j is needed by part i, and 0 otherwise. The objective is to obtain a matrix such as shown in Fig. 2, with the columns and rows arranged in an order corresponding to the groups identified by the CFP solution. The block diagonal matrix can then be used to determine the quality of the solution as specified by the chosen performance measure.

	M1	M2	M3	M4	M5	M6	M7	M8	M9
1	1			1					
2		1				1			
3			1		1		1		1
4	1			1				1	
5			1		1		1		
6	1			1				1	
7	1							1	
8		1				1			
9				1				1	
10			1		1				1
11		1				1			

Fig. 1. Machine-Part Incidence Matrix

	M5	M7	M9	M3	M1	M4	M8	M2	M6
10	1		1	1					
5	1	1		1					
3	1	1	1	1					
1					1	1			
4					1	1	1		
6					1	1	1		
9						1	1		
7					1		1		
2								1	1
8								1	1
11								1	1

Fig. 2. Machine-Part Incidence Matrix after Diagonalization

Several measures of goodness of machine-product groups in cellular manufacturing have been proposed. Chandrasekharan and Rajagopalan [20] introduced the first quantitative measure known as grouping efficiency. This measure allows for the normalized weighting of machine utilization and intercell traffic and is defined as fallow

Grouping Efficiency= $\eta = q\eta_1 + (1 - q)\eta_2$ (1)

Where

η_1= ratio of the number of 1´s in the diagonal blocks to the total number of elements in the diagonal blocks of the final matrix;

η_2= ratio of the number of 0´s in the off-diagonal blocks to the total number of elements in the off-diagonal blocks of the final matrix;

q = weight factor ($0 \leq q \leq 1$).

Work with this measure demonstrated that it had weak discriminating power. (i.e. the ability to distinguish good quality grouping from bad). For example, a bad solution with many 1´s in the off-diagonal blocks often shows efficiency figures around 75%. When the matrix size increases, the effect of 1´s in the off-diagonal blocks becomes smaller, and in some cases, the effect of inter-cell moves is not reflected in grouping efficiency.

To overcome this drawback, Kumar and Chandrasekharan [19] proposed a measure called grouping efficacy. Unlike grouping efficiency, grouping efficacy is not affected by the size of the matrix.
The grouping efficacy can be defined as

Grouping Efficacy= $\mu = \frac{N_1 - N_1^{out}}{N_1 + N_0^{in}}$ (2)
Where

N_1 = total number of 1´s in matrix A;

N_1^{out} = total number of 1´s outside the diagonal blocks;

N_0^{in} = total number of 0´s inside the diagonal blocks.

The grouping efficacy is able to incorporate both the within-cell machine utilization and the inter-cell movement. It has a high capability to differentiate between well-structured and ill-structured matrices (high discriminating power) and generates block diagonal matrices which are attractive in practice. On the other hand, it does not require a weight factor. Although other measures of grouping quality, such as grouping index [21], group capability index [22], and doubly weighted grouping efficiency [23], have been presented in the literature, grouping efficacy has been used as a benchmark measure for most algorithmic studies. Our study employs grouping efficacy as the solution quality measure as this allows for comparisons to be made between the algorithms developed in this study with those using the same test problems in previous research.

Mathematical Model

$$\max \quad \mu = \frac{N_1 - N_1^{out}}{N_1 + N_0^{in}} \tag{3}$$

s.t

$$\sum_{l=1}^{c} x_{il} = 1 \quad i = 1,2,...,m \tag{4}$$

$$\sum_{l=1}^{c} y_{jl} = 1 \quad j = 1,2,...,p \tag{5}$$

$$\sum_{i=1}^{m} x_{il} \geq 1 \quad l = 1,2,...,c \tag{6}$$

$$\sum_{j=1}^{p} y_{jl} \geq 1 \quad l = 1,2,...,c \tag{7}$$

$$x_{il}, y_{ji} \in \{0,1\} \; i = 1,2,...,m \;\; j = 1,2,...,p \;\; l = 1,2,...,c$$

Where

$$N_1^{out} = \sum_{i=1}^{m} \sum_{i=1}^{c} \sum_{j=1}^{p} a_{ij}.(1 - x_{il}.y_{jl}) \tag{8}$$

$$N_0^{in} = \sum_{i=1}^{m} \sum_{i=1}^{c} \sum_{j=1}^{p} (a_{ij} - 1).x_{il}.y_{jl} \tag{9}$$

m: the total number of machines
p: the total number of parts
c: the total number of cells
X [x_{il}] is an $m \times c$ cell membership matrix, where x_{il} =1 if machine i is in cell l and 0 otherwise
Y [y_{jl}] is a p×c cell membership matrix, where y_{jl} =1 if part j is in cell l and 0 otherwise
A [a_{ij}] is a m×p The machine-part incidence matrix, where a_{ij} = 1 if machine j is needed by part i, and 0 otherwise.
According to the above model, for a given machine–part incidence matrix A, the result we want to obtain is the number of cells c, machine–cell membership xil and part–cell membership yjl after cell formation.

Equation (3) is our objective function for our cell formation problem, which calculate efficacy grouping. Constraints (4) and (5) ensure that each machine and part can only be assigned into one cell. Constraints (6) and (7) ensure that each cell must contain at least one machines and one part, respectively. Constraint (8) calculates total number of 1´s outside the diagonal blocks and Constraint (9) calculates total number of 0´s inside the diagonal blocks.

Grouping Genetic Algorithm: In this section the proposed group genetic algorithm (GGA) for manufacturing cell formation will be presented in detail based on its four main elements which are:

- solution coding scheme (chromosome),
- fitness function,
- initial population generation,
- operators (selection, crossover, mutation).

Solution coding scheme: From the defined mathematical model, the part–cell membership y_{jl} can be concluded for a given machine–part incidence matrix a_{ij} and machine–cell membership x_{il}, thereby we must gain machine–cell membership x_{il}, or must group machines. Therefore in us chromosomes we only introduce the machine. Cell locations are identified with a vertical slash (|) that designates the cutoff for the machine numbers comprising the cell. Such As /236/54/178/ indicates that machines 2, 3, 6 are assigned to cell 1, machines 5, 4 are assigned to cell 2 and machines 1, 7, 8 are assigned to cell 3.

Fitness function: For proposed algorithm, the fitness function is as fallow:

$$Fitness = \frac{N_1 - N_1^{out}}{N_1 + N_0^{in}} \tag{10}$$

According to our data tests, the illusionary good fitness will mislead the CFP into a pointless solution as all machines are grouped into one cell or one in each cell. This problem does not seem more important than those encountered using general GAs, because the number of cells was predetermined. In our research, we give a penalty to the illusionary solutions (all machines are grouped into one cell or one in each cell):

$$Fitness_{[illusionary\ solutions]} = -\alpha \tag{11}$$

Where α is a positive number as large as it can be.
Initial population generation: The initial population is generated only once at the beginning and is called old population in the first generation of the genetic algorithm. The size of the initial population is defined by the user. Population sizes ranging from 20 to 60 individuals are common in the literature. A special procedure was developed to generate a random initial population that respects constraint 4 in model (each machine type must belong to only one cell) and constraint 6 in model

(each cell consists of one machines at least) of the mathematical model. Each chromosome of the initial population is created according to the following procedure:

Step 1. Generate the number of cells (c) randomly, where c is a random positive integer and $2 \le c \le \min\{m-1, p-1\}$

Step 2. For each cell, randomly select one machine from the set of machines. (This step ensures that each cell must contain at least one machine).

Step 3. Group the remaining $m-c$ machines into c cells randomly.

Selection operation: The selecting of selection operators is an important part in GAs. This part is independent of other parts in GAs and has no direct relation with the problem and with the fitness function, crossover operator and mutation operator used in Gas. There are many different selection operators presented by some researchers. For example, the elitist model, the expected value model, the tournament selection and etc. Here the Universal sampling method is adopted for selecting the good strings and the probability of selecting each string is calculated by.

$$SI = \frac{F(i)}{F_{total}} \tag{12}$$

F(i): Fitness value of the ith string

F_{total}: Total fitness value of all strings that is calculated

$$F_{total} = \sum_{i=1}^{n} F(i) \tag{13}$$

Crossover operation: The crossover operator produces children by exchanging information contained in the parents. The crossover operator takes two chromosomes selected and tries to mate them generating the individuals for the next generation. Crossover occurs according to the following steps:

Step 1. Select two crossing sites randomly, determining the crossing sections, which stand by the cell group in each of the parents (see figure 3(a)).

Step 2. Inject the contents of the crossing section of the second parent at the first crossing site of the first parent (see figure 3(b)).

Step 3. Eliminate all machines (underlined in figure 3(b)) now occurring twice in the first parent, and then eliminate the empty cell in the first parent (see figure 3(c)).

Step 4. Apply steps 2 and 3 to generate the second child.

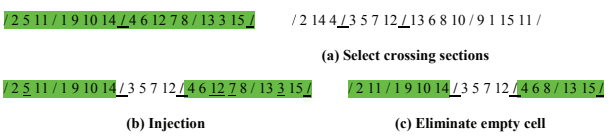

Fig 3. Crossover operator

It is possible that the new individuals violate the constraints of the mathematical model. There are basically two strategies to deal with this problem. The first one consists of penalizing the infeasible solutions in such a way that they will hardly propagate to the next generations. The second is to try to correct the chromosomes that violate the constraints. We adopted the second strategy as explained below.

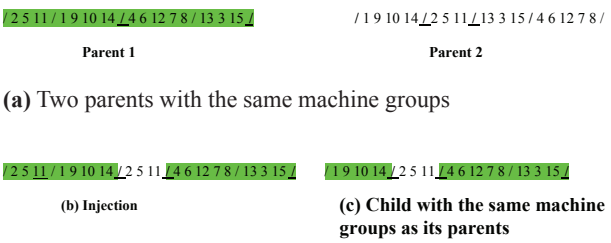

Fig 4. Inefficient Crossover operation.

One special case in our cell formation is shown in figure 4. If the machine sections of two parents are the same, no matter whether the cell sections of the two parents are the same or not, their offspring will always be the same as those two parents with the above crossover operator. We do not expect a phenomenon like simple cloning because it reduces the efficiency of our algorithm. In this case, we must compare the two selected parents. If machines are grouped into the same cells, we copy one of the parents as one child; another child is created randomly with the initialization operation.

Mutation operator: A mutation operator acts as a background operator, which is used to investigate some of the unvisited points in the search space and also to avoid premature convergence of the entire feasible space caused by some super chromosomes. Four mutation operators were developed for the cell formation problem. The first one exchanges genes between two groups. Following that, in each cell, a gene is randomly chosen and their values exchanged (Fig. 5a). The second mutation operator acts differently from the first operator. A number between 1 and (C-1) is randomly drawn. A direction (right or left) is also randomly selected and the frontier between the two adjacent groups is moved one machine in the selected direction (Fig. 5b). The Third mutation operator is a combination operator, which combine different cells into one cell (Fig. 5c). The Fourth mutation operator is a division operator, which divide one cell into two cells (Fig. 5d).

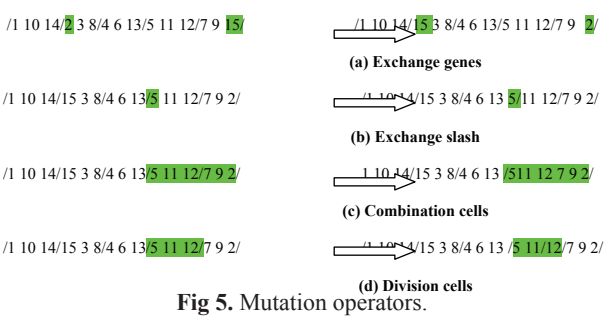

Fig 5. Mutation operators.

RESULTS AND DISCUSSION

In order to demonstrate our method, we first introduce a simple CFP example, and then to demonstrate the performance of the proposed algorithm tested the grouping genetic algorithm on 14 GT instances collected from the literature. We compiled our algorithm in Matlab7.0 and all the numerical examples were tested on a PC with an Intel Pentium 4 2GHz processor, 1GHz memory and Winxp Professional Operating System. We show that our method is capable of dealing with CFP more efficiently and flexibly without predetermining the number of cells.

A simple example: The simple example is a CFP with five machines and eight parts, and the result after cell formation is shown in Fig. 6(b). The total number of cells is two for the optimal solution. The performance of the simple CFP is shown in Fig 7. and the parameters are given as

$P_Z = 10, P_C = .7, P_M = .3, T_{end} = 25$.

From the figure of performance, the optimal solution of Fig 7. is found in the sixth generation, and the CPU time is 0.4250 s. The initial population and end population are listed in table 1.

Table 1. Sample population of the simple CFP example

NO.	Initial population		End population	
	Chromosome	μ	Chromosome	μ
1	/45/132/	0.66	/1/45/32/	0.78
2	/5/23/4/1/	0.68	/541/32/	1
3	/2/45/13/	0.55	/123/45/	0.66
4	/345/12/	0.4	/45/23/1/	0.78
5	/23/45/1/	0.78	/32/145/	1
6	/34/125/	0.4	/4/5/1/23/	0.68
7	/3/5/12/4/	0.39	/32/4/51/	0.62
8	/12/345/	0.4	/32/154/	1
9	/12/5/3/4/	0.39	/45/1/23/	0.78
10	/3/25/1/4/	0.32	/4/5/123/	0.66

Fig 7. Example of the performance of a simple CFP

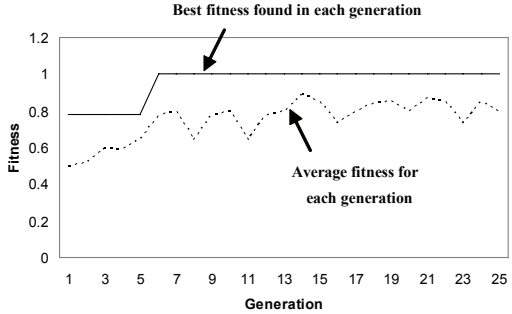

Table 2. Selected GT problems from the literature

NO.	Source	Size
1	King and Nakornchai (1982)	5×7
2	Seiffodini (1989)	5×18
3	Kusiak (1992)	6×8
4	Chandrasekharan and Raja. (1986a)	8×20
5	Chandrasekharan and Raja. (1986b)	8×20
6	King (1980)	16×43
7	Chandrasekaran and Raja. (1989)-1	24×40
8	Chandrasekaran and Raja. (1989)-2	24×40
9	Chandrasekaran and Raja. (1989)-3	24×40
10	Chandrasekaran and Raja. (1989)-5	24×40
11	Chandrasekaran and Raja. (1989)-6	24×40
12	Chandrasekaran and Raja. (1989)-7	24×40
13	King and Nakornchai (1982)	36×90
14	Chandrasekaran and Raja. (1987)	40×100

0	1	2	3	4	5	6	7	8
1	1	0	0	1	0	0	1	0
2	0	1	1	0	1	1	0	1
3	0	1	1	0	1	1	0	1
4	1	0	0	1	0	0	1	0
5	1	0	0	1	0	0	1	0

a) 5×8 CFP

0	1	4	7	2	3	5	6	8
1	1	1	1	0	0	0	0	0
4	1	1	1	0	0	0	0	0
5	1	1	1	0	0	0	0	0
2	0	0	0	1	1	1	1	1
3	0	0	0	1	1	1	1	1

5×8 CFP after grouped

before grouped (b)

Fig 6. Simple CFP before and after Grouping.

The test results are presented in Table 3. As can be seen in Table 3, the algorithm proposed in this paper obtained machine/product groupings, which have a grouping efficacy that is never smaller than any of the best reported results. More specifically, the algorithm obtains, for 7 (50%) problems, values of the grouping efficacy that are equal to the best ones found in the literature and improves the values of the grouping efficacy for 7 (50%) problems. On the other hand, proposed GGA can solve the CFP without predetermination of the number of cells.

Table 3. Test Results

NO.	ZODIAC	GRAFIC	Cheng et al.	Proposed Algorithm	Improvement
1	73.68	73.68	-	73.68	0.00%
2	77.36	-	77.36	78.54	1.53%
3	76.92	-	76.92	76.92	0.00%
4	85.24	85.24	85.24	85.25	0.01%
5	58.33	58.13	58.72	58.72	0.00%
6	53.76	54.39	53.89	54.47	0.15%
7	100	100	100	100	0.00%
8	85.11	85.11	85.11	85.11	0.00%
9	73.51	73.51	73.51	73.51	0.00%
10	20.42	43.27	49.37	49.54	0.34%
11	18.23	44.51	44.67	45.42	1.90%
12	17.61	41.67	42.5	43.21	1.67%
13	32.73	39.41	40.05	43.71	9.14%
14	83.86	8392	84.03	84.03	0.00%

Conclusion

In this article, we have presented a more efficient and flexible method for solving the cell formation problem (CFP) by adopting method of the grouping genetic algorithm (GGA) that is also an extension of GAs. Computational experience with the algorithm, on a set of 14 GT problems from the literature, has shown that it performs remarkably well and the obtained solutions are at least as good as the ones found the literature. On the other hand, the numerical examples and comparisons show that the CFP can be optimized with respect to both grouping machines into cells and deciding on the number of cells using our method. However, once we are grouping machines, the number of parts has no effect on the size of the space solution making the algorithm attractive for problems where the number of parts is large.

In our research, the effort to improve the representation of GAs and their operators is also encouraged. The proposed GGA algorithm has several new features such as the chromosome codification scheme, correction mechanism and the crossover and mutation operators that work directly with the group of machines as opposed to individual machines. Some of Further research projects that can be conducted are: (i) establishing a strategy that generates an initial population with higher score. For example, Better results may be obtained if similarity coefficients are used to calculate the similarity between the machine and the groups and assign the machine to the group that results in the highest similarity coefficient value; (ii) Different strategies in the choice of the selection operator. For example, a tournament operator can be used. We can search different selection operators and choice an operator that gives a good solution with a shorter time; (iii) modifying the constraints to consider any condition about machines and grouping of them; (iv) modifying the fitness function to consider important design factors such as processing time, production requirements and available time on machine in a given period, etc.

Acknowledgement

This research is sponsored in part University of Bojnourd

REFERENCES

[1] Shunk DL. 1985. Group technology provides organized approach to realizing benefits of CIMS. J. Industrial Engineering. 17: 74-81.

[2] Burbidge J.L. 1963. Production flow analysis. The Production Engineer. 42: 742–752.

[3] Offodile F.O., Mehrez A., Grznar J. 1994. Cellular manufacturing: a taxonomic review framework. J. Manuf. Syst., 13: 196–220.

[4] Holland, J.H. 1975. Adaptation in Natural and Artificial Systems. University of Michigan Press: Ann Arbor. MI.

[5] Goldberg, D. 1989. Genetic algorithms in search, optimization, and machine learning. Reading, PA: Addison-Wesley.

[6] Falkenauer, E. 1998. Genetic Algorithms and Grouping. Wiley. New York.

[7] Venugopal V., Narendran T.T. 1992. A genetic algorithm approach to the machine–component grouping problem with multiple objectives. Comp. & Indus. Engin., 22: 469–480. 1992

[8] Billo R.E., Bidanda B., Tate D. 1996. A genetic cluster algorithm for the machine–component grouping problem. J. Intell. Manuf., 7: 229–241.

[9] Joines J.A., Culberth C.T., King R.E. 1996. Manufacturing cell design: an integer programming model employing genetic algorithms. IIE Transactions. 28: 69–85.

[10] Cheng C.H., Gupta Y.P., Lee W.H., Wong K.F. 1998. A TSP-based heuristic for forming machine groups and part families. Int. J. Prod. Res. 36: 1325–1337.

[11] Mak K.L., Wong Y.S., Wang X.X. 2000. An adaptive genetic algorithm for manufacturing cell formation. Int. J. Adv. Manuf. Technol. 16: 491–497.

[12] Rao H.A., Pham S.N., Gu P. 1999. A genetic algorithm-based approach for design of manufacturing systems: an industrial application. Int. J. Prod. Res. 37: 557–580.

[13] Zhao C.W., Wu Z.M., 2000. A genetic algorithm for manufacturing cell formation with multiple routes and multiple objectives. Int. J. Prod. Res. 38: 385–395.

[14] Onwubolu G.C. Mutingi M. 2001. A genetic algorithm approach to cellular manufacturing systems. Comp. & Ind. Engin. 39:125–144.

[15] Uddin M.K., Shanker K., 2002. Grouping of parts and machines in presence of alternative process routes by genetic algorithm. Int. J. Prod. Econ. 76: 219–228.

[16] De Lit P., Falkenauer E., Delchambre A. 2000. Grouping genetic algorithms: an efficient method to solve the cell formation problem. Math. Comp. Simul. 51: 257–271.

[17] Eduardo Vila G.F. and Alexandre J.T. 2006. A group genetic algorithm for the machine cell formation problem. Int. J. Production Economics. 102: 1–21. [18] Tabitha L.J., Evelyn Brown C., Kellie B.K. 2007. A hybrid grouping genetic algorithm for

[18] Tabitha L.J., Evelyn Brown C., Kellie B.K. 2007. A hybrid grouping genetic algorithm for the cell formation problem. Computers & Operations Research. 34: 2059–2079.

[19] Kumar K.R. Chandrasekharan M.P. 1990. Grouping efficacy: A quantitative criterion for goodness of block diagonal forms of binary matrices in group technology. International Journal of Production Research. 28(2): 233-243.

[20] Chandrashekharan M. P., Rajagopalan R. 1986. An ideal seed non-hierarquical clustering algorithm for cellular manufacturing. International Journal of Production Research. 24(2): 451-464.

[21] Seifoddini H. Djassemi M., 1996. The threshold value of a quality index for formation of cellular manufacturing systems. International Journal of Production Research. 34: 3401–3416.

[22] Sarker B. and Mondal S. 1999. Grouping efficiency measures in cellular manufacturing: a survey and critical review. International Journal of Production Research. 37: 285–314.

[23] Sarker B. 2001. Measures of grouping efficiency in cellular manufacturing systems. European Journal of Operational Research. 130: 588–611.

[24] Chandrasekharan M.P., Rajagopalan R. 1987. ZODIAC-An algorithm for concurrent formation of part families and machine cells. International Journal of Production Research. 25(6): 835-850.

[25] Srinivasan G., Narendran T.T. 1991. GRAFICS-A nonhierarchical clustering algorithm for group technology. International Journal of Production Research. 29(3): 463-478.