A Fuzzy Particle Swarm Optimization Algorithm for a Cell Formation Problem

Esmaeil Mehdizadeh[1], Reza Tavakkoli-Moghaddam [2]

1Faculty of Industrial and Mechanical Engineering, Islamic Azad University, Qazvin Branch, Iran
2 Department of Industrial Engineering, College of Engineering, University of Tehran, Tehran, Iran
Email: mehdizadeh@qazviniau.ac.ir

Abstract— *Group technology (GT) is a useful way to increase productivity with high quality in cellular manufacturing systems (CMSs), in which cell formation (CF) is a key step in the GT philosophy. When boundaries between groups are fuzzy, fuzzy clustering has been successfully adapted to solve the CF problem; however, it may result uneven distribution of parts/machines where the problem becomes larger. In this case, particle swarm optimization (PSO) can be used to tackle such a hard problem. This paper proposes a hybrid algorithm based on the fuzzy clustering and particle swarm optimization (FPSO) to solve the given CF problem. We experiment a number of examples to show the efficiency of the proposed algorithm and find that our proposed FPSO algorithm is able to obtain good results at reasonable time.*

Keywords— Cellular manufacturing systems, cell formation, fuzzy clustering, particle swarm optimization (PSO)

1 Introduction

Cell formation (CF) is a key step in Group technology (GT) that is used to design a good cellular manufacturing system by using the similarities of parts related to machines so that it can identify part families and machine groups. The parts in the same machine group have similar requirement so that GT can reduce travel and setup time. In CF the part/machine matrix, which has $m \times p$ dimensions with binary components, is usually described and given. The m rows indicate machines and p columns represent p parts that need to be operated upon. In matrix m, ‘‘1’’ (‘‘0’’) represents that this part should be (not) worked on the machine. The matrix exhibits parts requirement relative to machines. Our objective is to group parts and machines just like a cell.

Cell formation problem (CFP) is shown to be NP-hard [1] in the strong sense, and obtaining an optimal solution for the large-sized problems in reasonable computational time is extremely difficult.

Clustering involves the task of dividing data points into homogeneous classes or clusters so that items in the same class are as similar as possible and items in different classes are as dissimilar as possible. In real applications there is very often no sharp boundary between clusters so that fuzzy clustering is often better studied for the data. Membership degrees between zero and one are used in fuzzy clustering instead of crisp assignments of the data to clusters. In fuzzy clustering, the data points can belong to more than one cluster, and associated with each of the points are membership grades which indicate the degree to which the data points belong to the different clusters. In deterministic CF methods assumed well- defined boundaries between part–machine cells. These crisp boundary assumptions may fail to fully describe the case where the part–machine cell boundaries are fuzzy. This is why fuzzy clustering algorithms were applied for CF.

There are many CF methods in the literature [2]. The CF models can also be categorized into those of crisp or fuzzy. Crisp models assume that there are well-defined boundaries between groups and therefore assign each part or machine to only one family. In reality, some parts may belong to one part family, but there may have parts whose linkages are much less evident.

Various Clustering methods have been proposed to solve the CF problem. The fuzzy *c*-means (FCM) algorithm was first used in part-family formation by Xu and Wang [3]. FCM algorithm performs well with small and well-structured data sets. However, when the data set becomes larger, the algorithm may result in clustering errors, infeasible solutions, and uneven distribution of parts/machines. Then, several researchers have proposed alternatives to remedy these weaknesses with mixed success. For example, Chu and Hayya [4] improved the study carried out by Xu and Wang [3].

Al- Ahmari [5], Yang et al. [6] and Feng et al. [7] applied concepts of fuzzy clustering on the cell formation problem. Li et al. [8] improved fuzzy clustering algorithm to overcome the deficiencies of FCM. Since large instances are so difficult to optimally solve, approximate methods are needed. Perhaps, meta-heuristics are the most successful approximate methods that have been used so far. Thus, for example, Boctor [9] and Chen and Srivastava [10] used simulated annealing. Genetic algorithms have been used by Kazerooni et al. [11], Brown and Sumichrast [12], Ravichandran et al. [13]. Aljaber et al. [14] and Lozano et al. [15] applied tabu search. Attila [16] proposed an ant system algorithm. Zhao et al. [17] used swarm intelligence, and finally Andres and Lozano [18] presented particle swarm optimization (PSO) algorithm to solve the cell formation problem addressed in GT.

In the previous work, fuzzy clustering and PSO have been applied in the CF in separate. Our aim is to design a fuzzy particle swarm optimization (FPSO) clustering algorithm to solve the part–machine grouping problem,

ISBN: 978-989-95079-6-8

which is known as a hard combinatorial problem.

The rest of this paper is given bellow. The fuzzy cell formation model is presented in Section 2. In Section 3, the PSO algorithm is presented. The proposed FPSO algorithm is presented in Section 4. Computational results with a number of test problems taken from the literature are shown in Section 5. Finally, Section 6 draws conclusions, suggests directions for future research and discusses the limitations of the research.

2 Fuzzy clustering problem

The fuzzy cell formation (FCF) problem described by Li et al. [8] as follows: Given the routing information of n parts and m machines, the goal of CF is to cluster the parts into c part families and the corresponding machines into machine cells. The classification result can be expressed as a matrix $U=[\mu_{ik}]_{c\times n}$, (k= 1, 2,…, n and i= 1, 2,…, c), and μ_{ik} is the membership degree of part k to group i, which satisfies:

$$0 \le \mu_{ik} \le 1 \tag{1}$$

$$\sum_{i=1}^{c} \mu_{ik} = 1 \tag{2}$$

$$0 < \sum_{k=1}^{n} \mu_{ik} \le n \tag{3}$$

The FCM clustering algorithm is based on the minimization of the following equation:

$$J(p) = \sum_{k=1}^{n} \sum_{i=1}^{c} [\mu_{ik}]^m \|x_k - V_i\|^2 \tag{4}$$

where, m>1 is a real number governing the influence of membership grades, V_i is the cluster center of the part family i, and x_k is the vector of part k. The necessary conditions for minimizing $J(p)$ are the following update equations:

$$V_i = \frac{\sum_{k=1}^{n} [\mu_{ij}]^m x_k}{\sum_{k=1}^{n} [\mu_{ij}]^m} \tag{5}$$

which, c= 1, 2, … , n.

$$\mu_{ik}^{(t+1)} = \left[\sum_{j=1}^{c} \left(\frac{\|x_k - V_i^{(t)}\|^2}{\|x_k - V_j^{(t)}\|^2} \right)^{\frac{1}{m-1}} \right]^{-1} \tag{6}$$

where, $\|x_k - V_i\|^2$ represents the Euclidean distance between x_k and V_i , and $\mu_{ik}^{(t+1)}$ is the membership degree of part k in group i.

Many variations of FCM algorithms can be found in Bezdek [19]. The algorithm is based on the assumption that the desired number of clusters c, real number m, stopping criterion ε, and a particular distance are given.

Step 1: Let t=0 and select an initial fuzzy pseudo-partition $p^{(0)}$.

Step 2: Calculate c cluster centers, $V_1^{(t)}$, …, $V_c^{(t)}$, by (5) for $p^{(t)}$ and the chosen value of m.

Step 3: Define $\mu_i^{(t+1)}$ by (6) and update $p^{(t+1)}$

Step 4: Compare $p^{(t)}$ and $p^{(t=1)}$. If $\left| p^{(t+1)} - p^{(t)} \right| \le \varepsilon$, then stop the algorithm; otherwise, increase t by one and then return to Step 2.

Since the fuzzy clustering problem is a combinatorial optimization problem that is hard to solve [20]. Large instances are so difficult to optimally solve, approximate methods are then needed.

3 Particle swarm optimization

In particle swarm optimization (PSO) a number of simple entities—the particles—are placed in the search space of some problem or function, and each evaluates the objective function at its current location. Each particle then determines its movement through the search space by combining some aspect of the history of its own current and best (best-fitness) locations with those of one or more members of the swarm, with some random perturbations. The next iteration takes place after all particles have been moved. Eventually the swarm as a whole likes a flock of birds collectively foraging for food is likely to move close to an optimum of the fitness function.

There are many variants of the PSO proposed in the literature so far, when Eberhart and Kennedy [21] first introduced this technique. A version of this algorithm is used for part-machine grouping by Andres and Lozano [18].

For description of The PSO algorithm, first, let me define the notation adopted in this paper: the i-th particle of the swarm is represented by the D-dimensional vector $x_i=(x_{i1}, x_{i2}, \ldots, x_{iD})$ and the best particle of the swarm (i.e., the particle with the smallest function value) is denoted by index p_g. The best previous position (i.e., the position giving the best function value) of the i-th particle is recorded and represented $p_i=(\rho_{i1}, \rho_{i2}, \ldots, \rho_{iD})$, and the position change (velocity) of the i-th particle is $Vel_i=(Vel_{i1}, Vel_{i2}, \ldots, Vel_{iD})$. The particles are then manipulated according to the following equations:

$$Vel_{id}(t+1) = \chi\{wVel_{id}(t) + c_1\phi_1[\rho_{id}(t) - x_{id}(t)] + c_2\phi_2[(\rho_{gd}(t) - x_{id}(t)]\} \tag{7}$$

$$x_{id}(t+1) = x_{id}(t) + Vel_{id}(t+1) \tag{8}$$

where, d=1, 2, …, D ; i=1, 2, …, n; and n is the size of the swarm; w is the inertia weight; c_1 and c_2 are two positive acceleration constants; ϕ_1 and ϕ_2 are two random values into the range [0, 1]; χ is a constriction factor that is used in constrained optimization problems in order to control the magnitude of the velocity (in unconstrained optimization problems it is usually set to 1.0).

4 Proposed FPSO algorithm

In the fuzzy clustering, a single particle represents a cluster center vector and a swarm represents a number of candidates clustering for the current data vector. Here; each point or data vector belongs to every various cluster by different membership function, thus; assign a fuzzy membership to each point or data vector. Each cluster has a cluster center, and per iteration presents a solution that gives

ISBN: 978-989-95079-6-8

a vector of cluster centers. We determine the position of each vector for every particle and update it, then change the position of cluster centers based of particles. For the purpose of our algorithm, we define the following notations:

n	Number of part
c	Number of cluster center
$V_l^{(t)}$	Position of l-th particle' cluster center in stage t
$Vel_l^{(t)}$	Velocity of l-th particle in stage t
x_k	Vector of parts ($k = 1,2,.....,n$)
$\rho_l^{(t)}$	Best position funded by l-th particle in stage t
$\rho_g^{(t)}$	Best position funded by all particles in stage t
$P^{(t)}$	Fuzzy pseudo partition in stage t
$\mu_{ik}^{(t)}$	Membership function of k-th part in stage t into i-th cluster

The fitness of particles is easily measured by (4). The c-means algorithm tends to converge faster than the proposed FPSO algorithm with a less accurate clustering. In this section, the performance of the PSO clustering algorithm is improved by seeding the initial swarm with the result of the c-means algorithm. The FPSO algorithm first executes the c-means algorithm once. In this case, the c-means clustering algorithm is terminated by one of two stopping criteria: I) the maximum number of iterations; or II) $\left|p^{(t+1)} - p^{(t)}\right| \le \varepsilon$. The result of c-means algorithm is then used as one of the particles, while the rest of the swarms are initialized randomly. The following algorithm can use to finding cluster for each data vector or part:

Step 1: Let t=0, select initial parameters such as number of cluster center c, initial position of particle by the FCM, initial velocity of particles, c_1, c_2, w, χ, and a real number $m \in (1, \infty)$, and a small positive number ε for stopping criterion.
Step 2: Calculate $\mu_{ik}^{(t)}$ for all particles and all i=1, 2, …, c and k=1, 2, …, n by (6) and update $p^{(t+1)}$.
Step 3: For each particle, calculate the fitness by using (4).
Step 4: Update the global best and local best position.
Step 5: Update $Vel_l^{(t)}$ and $V_l^{(t)}$ for all l=1, 2, …, *n_particle* by using (7) and (8).
Step 6: Update $p^{(t+1)}$ by the Step 2.
Step 7: Compare $p^{(t)}$ and $p^{(t+1)}$. If $\left|p^{(t+1)} - p^{(t)}\right| \le \varepsilon$, then stop; otherwise, increase t by one and continue form Step 3.

5 Numerical example

In this section, examples from the literature are considered to illustrate the application of the proposed fuzzy algorithm in the cell formation problem. We compare the results of the FCM and FPSO algorithms on various problems taken from the literature. Their performances are measured by the objective function value given in (4) and CPU time. A general rule of thumb is that a clustering result with lower *J(p)* and lower CPU time is preferable. For a comparable assessment, we code these methods by using the fuzzy tools available in MATLAB 7 and the FPSO, respectively, with 10 particles, w=0.72, and $c_1 = c_2 = 1.49$. For our experimental tests, we use a PC Pentium III, CPU 1133 MHz and 256 MB of RAM for the same parameters for all algorithms implementations: m=2, the maximum number of iterations is 100 and $\varepsilon = 0.00001$.

Table 1: Data from Chu and Hayya [4]

	Parts								
Machines	1	2	3	4	5	6	7	8	9
1	1	1	0	0	1	0	0	0	0
2	1	1	0	0	0	1	0	0	1
3	0	0	1	0	0	0	1	1	0
4	0	1	1	1	0	0	0	1	0
5	1	0	0	1	1	0	0	1	0
6	0	1	0	0	0	1	0	0	1
7	0	0	1	0	0	0	1	1	0
8	0	0	1	1	1	0	1	1	0
9	0	1	0	0	0	1	0	0	1

Table 2: The membership matrix for cells for first example

Machines	Memberships for cells		
	1	2	3
1	.2239	.2755	.5006
2	.0912	.7838	.1250
3	.9335	.0249	.0417
4	.4413	.1871	.3716
5	.0712	.0392	.8895
6	.0160	.9666	.0174
7	.9335	.0249	.0417
8	.5359	.0947	.3694
9	.0160	.9666	.0174

Table 3: The membership matrix for part families for first example

Parts	Memberships for part families		
	1	2	3
1	.7110	.1656	.1234
2	.2502	.6234	.1264
3	.0698	.0441	.8861
4	.5026	.1370	.3603
5	. 7659	.0922	.1419
6	.0437	.9346	.0216
7	.2061	.1233	.6706
8	.0751	.0330	.8919
9	.0437	.9346	.0216

Table 4: Comparison of Chu and Hayya's approach and FPSO approach

	Chu and Hayya's approach		The FPSO approach	
	Machine cell	Part families	Machine cell	Part families
Cell 1	M1,M5	P1,P4,P5	M1,M5	P1,P4,P5
Cell 2	M2,M6,M9	P2,P6,P9	M2,M6,M9	P2,P6,P9
Cell 3	M3,M4 M7,M8	P3,P7,P8	M3,M4 M7,M8	P3,P7,P8
$J(p)$	3.729600	3.859139	3.729600	3.859139

ISBN: 978-989-95079-6-8

The first example taken from Chu and Hayya [4] consists of nine machines and nine parts is illustrated in Table 1. Tables 2 and 3 show the membership matrix values for cells and part families, respectively. The final membership matrix values indicates the degree of membership of each machine associated with the machine cell (MC) and can be configured as: MC (1) = {M1, M5}, MC (2) = {M2, M6, M9}, and MC (3) = {M3, M4, M7, M8}. Similarity, the part family (PF) can be configured as: PF (1) = {P1, P4, P5}, PF (2) = {P2, P6, P9}, and PF (3) = {P3, P7, P8}. Table 4 compares Chu and Hayya's approach with the FPSO approach results. As illustrated in the table, no changes are observed between machine cells and part families.

Table 5: Data from Susanto et al. [22]

Machine	Parts									
	1	2	3	4	5	6	7	8	9	10
1	1	1	1	1	0	0	0	0	0	0
2	1	1	1	1	0	0	0	0	0	0
3	1	1	1	0	0	0	0	0	0	0
4	1	1	0	0	0	0	0	0	0	0
5	0	0	0	0	1	1	1	0	0	0
6	0	0	0	0	1	1	1	0	0	0
7	0	0	0	1	1	1	0	0	0	0
8	0	0	0	0	0	0	0	1	1	1
9	0	0	0	0	0	0	0	1	1	1
10	0	0	0	0	0	0	0	1	1	0
11	1	1	0	0	1	1	0	0	0	0
12	1	1	0	0	1	1	0	0	0	0

The second example is taken from Susanto et al. [22] as shown in Table 5. The obtained results are illustrated in Tables 6 and 7 and can be configured as: MC (1) = {M5, M6, M7, M11, M12}, MC (2) = {M8, M9, M10}, and MC (3) = {M1, M2, M3, M4}, $J_{(p)} = 4.844414$. Part families are as follows: PF (1) = {P5, P6, P7}, PF (2) = {P8, P9, P10}, and PF (3) = {P1, P2, P3, P4}, J (p) = 5.020256.

Table 6: The membership matrix for cells for second example

Machines	Degree of membership		
	1	2	3
1	.0788	.0701	.8511
2	.0788	.0701	.8511
3	.0842	.0730	.8428
4	.1233	.1082	.7685
5	.9017	.0460	.0522
6	.9017	.0460	.0522
7	.6359	.1578	.2062
8	.0541	.8940	.0519
9	.0541	.8940	.0519
10	.01159	.7661	.1180
11	.4311	.1561	.4128
12	.4311	.1561	.4128

Table 7: The membership matrix for part families for second example

Parts	Degree of membership		
	1	2	3
1	0.9265	0.0315	0.0420
2	0.9265	0.0315	0.0420
3	0.6430	0.1975	0.1595
4	0.4342	0.2743	0.2916
5	0.0238	0.0236	0.9525
6	0.0238	0.0236	0.9525
7	0.1765	0.2902	0.5333
8	0.0218	0.9534	0.0248
9	0.0218	0.9534	0.0248
10	0.0702	0.8483	0.0815

Table 8: Solution to Al-Ahmari

Cell	Machine Cells	$J(p)$ FCM	CPU Time FCM	$J(p)$ FPSO	CPU Time FPSO
2	{1,3,4,9,10,13,14,16,17,20,21,22},{2,5,6,7,8,11,12,15,18,19,23,24}	54.583351	0.300	54.583351	0.190
3	{1,3,13,16,20,21,22},{6,7,9,10,14,17,23,24},{2,4,5,8,11,12,15,18,19}	36.388919	0.641	36.388919	0.170
4	{2,3,4,6,9,10,11,12,15,16,17,18},{1,8,14,20,23,24},{7,13,21,22},{5,19}	22.515392	0.490	22.515392	0.431
5	{1,13,21,22},{2,5,11,19},{3,4,7,14,16,20,23,24},{9,10,17},{6,8,12,15,18}	14.691166	0.852	14.691166	0.280
6	{3,20},{1,13,21,22},{2,5,11,19},{7,14,23,24},{4,6,8,12,15,16,18},{9,10,17}	9.046538	0.571	9.046538	0.350
7	{4,16},{9,10,17},{6,8,12,15,18},{3,20},{2,5,11,19},{1,13,21,22},{7,14,23,24}	6.172838	0.611	6.172838	0.581
8	{1,13,21,22},{7,23,24},{4,16},{14},{9,10,17},{6,8,12,15,18},{3,20},{2,5,11,19}	4.139911	0.651	4.139911	0.591
9	{9,10},{14},{17},{6,8,12,15,18},{3,20},{2,5,11,19},{1,13,21,22},{7,23,24},{4,16}	2.772339	0.671	2.772339	0.411
10	{4,16},{6,7,23,24},{17},{8,12,15,18},{2,5,11,19},{1,14},{13,21,22},{6},{3,20},{9,10}	1.694722	0.551	1.694722	0.541

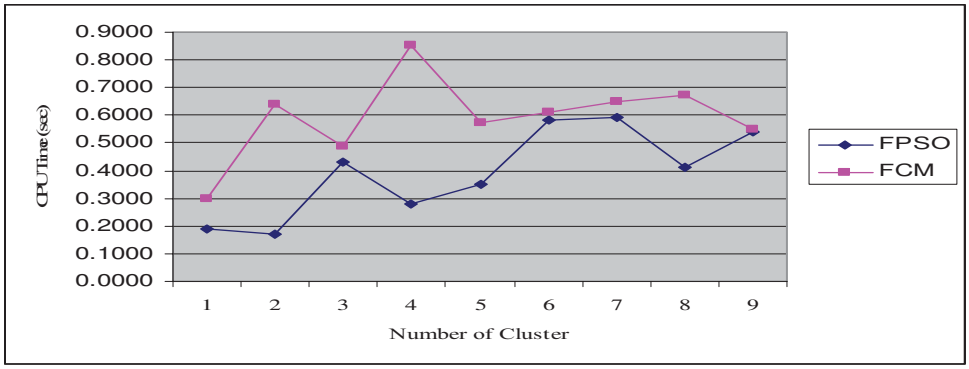

Figure 1: CPU Time comparison between FPSO and FCM algorithms

ISBN: 978-989-95079-6-8

The proposed approach is also tested on a large-sized problem taken from Al-Ahmari [5]. In the example, the problem with 24 machines and 40 parts is considered. Different numbers of manufacturing cells are used (i.e., 2, 3, 4, 5, 6, 7, 8, 9, and 10). The results summarized in table 8 and Figure 1. This example demonstrates the possibility of using the approach for large-scale CF problems, and evaluates the obtained results using $J(P)$.

6 Conclusion

Group technology (GT) is a useful way to increase productivity with high quality in flexible manufacturing systems; and CF is a key step in GT. It is used to design a good cellular manufacturing that uses the similarities of parts related to machines so that it can identify part families and machine groups. The cell formation problem is NP-complete and different -heuristic methods have been used to solve it. Particle swarm optimization is one of them. On the other hand, the crisp models fail to fully reflect the complex nature of part features or routing data, where boundaries between groups are fuzzy. Fuzzy clustering has been successfully adapted to solve the cell formation problem, but when the problem becomes larger, the fuzzy clustering algorithms may result uneven distribution of parts/machines. In the previous works, fuzzy clustering and particle swarm optimization have been applied in CF individually. We have designed a fuzzy particle swarm optimization clustering algorithm (FPSO) to solve the part–machine grouping problem, which is a hard combinatorial problem. The presented numerical examples confirm the effectiveness of the proposed approach. It is found that this algorithm provides a good solution to CF problems at reasonable time and allows the user in formulating the required size of machine cells and part families.

References

[1] A. Ballakur and H.J. Steudel, "A Within Cell Utilization Based Heuristic for Designing Cellular Manufacturing Systems", *Int. J. of Prod. Res.*, 25: 639-55, 1987.

[2] Y. Yin and K. Yasuda, "Similarity Coefficient Methods Applied to the Cell Formation Problem: A Taxonomy and Review", *Int. J. Prod. Economics,* 101: 329-352, 2006.

[3] H. Xu and H.P. Wang, "Part Family Formation for GT Applications Based on Fuzzy Mathematics", *Int. J. of Prod. Research,* 27: 1637-1651, 1989.

[4] C.H. Chu and J.C. Hayya, "A Fuzzy Clustering Approach to Manufacturing Cell Formation", *Int. J. of Prod. Research*, 29: 1475-1487, 1991.

[5] A.M.A. Al-Ahmari, "A Fuzzy Analysis Approach for Part-Machine Grouping in Cellular Manufacturing Systems", *Integrated Manufacturing System,* 13:7: 489-497, 2002.

[6] M.S. Yang, W.L. Hung and F.C. Cheng, "Mixed-Variable Fuzzy Clustering Approach to Part Family and Machine Cell Formation for GT Applications", *Int. J. Prod. Economics,* 103: 185-198, 2006.

[7] C.Y. Chen and F. Ye, "Adaptive Hyper-Fuzzy Partition Particle Swarm Optimization Clustering Algorithm", *Cybernetics and Systems,* 37:463-479, 2006.

[8] J. Li, C.H Chu, Y. Wang and W. Yan, "An Improved Fuzzy Clustering Method for Cellular Manufacturing*", Int. J. of Prod.. Research,* 45: 1049-1062, 2007.

[9] F.F. Boctor, "A Linear Formulation of the Machine-Part Cell Formation Problem", *Int. J. Prod. Res.,* 29: 343-56, 1991.

[10] W. Chen and B. Srivastava, "Simulated Annealing Procedures for Forming Machine Cells in Group Technology", *Eur. J. Oper. Res.*, 75: 100-111, 1994.

[11] M. Kazerooni, L. Luong and K. Abhary," Cell Formation using Genetic Algorithms. *Int. J. of Flexible Autom. Integr Manuf*, 3: 283-299, 1995.

[12] E.C. Brown and R.T. Sumichrast, "A Grouping Genetic Algorithm for the Cell Formation Problem", *Int. J. Prod. Res.,* 39: 3651-3669, 2001.

[13] K.S. Ravichandran, K. Chandra, S. Rao and R. Saravanan, "The Role of Fuzzy and Genetic Algorithms in Part Family Formation and Sequence Optimization for Flexible Manufacturing Systems", *Int. J. of Adv. Manuf. Technology,* 19: 879-888, 2002.

[14] N. Aljaber, W. Baek and C. L. Chen, "A Tabu Search Approach to the Cell Formation Problem", *Computers and Industrial Eng.*, 32: 169-185, 1997.

[15] S. Lozano, B. Adenso-Dı, I. Eguia, and L. Onieva, "A One-Step Tabu Search Algorithm for Manufacturing Cell Design", *J. of Oper. Res. Soc.,* 50: 509-516, 1999.

[16] A.A. Islier, "Group Technology by an Ant System Algorithm", *Int. J. of Prod. Research,* 43: 913-932, 2005.

[17] G. Zhao, P. Jiang, M. Zheng, G. Zhou, and B. Chen, "Using Swarm Intelligence Algorithm to Solve a Part Clustering Model Based on Weighted Directed Graph", *Int. J. of Prod. Res.,* 2007. A. Andres and S. Lozano, "A Particle Swarm Optimization Algorithm for Part–Machine Grouping", *Robotics and Computer-Integrated Manufacturing,* 22: 468-474, 2006.

[18] J.C. Bezdek, "*Pattern Recognition with Fuzzy Objective Function Algorithms*", Plenum Press: New York, 1981.32

[19] J.G. Klir and B. Yuan, "*Fuzzy Sets and Fuzzy Logic, Theory and Applications"*, Prentice Hall of India, 257-262, 2003.

[20] J. Kennedy and R.C. Eberhart, "Particle Swarm Optimization", In: *Proceedings of IEEE Int. Conf. on Neural Networks,* Perth, Australia, 1942-1948, 1995.

[21] S. Susanto, R.D. Kennedy and J.W.H. Price, "A New Fuzzy-*C*-Means and Assignment Technique-Based Cell Formation Algorithm to Perform Part-Type Clusters and Machine-Type Clusters Separately", *Prod. Planning & Control,* 10: 375-388, 1999.

ISBN: 978-989-95079-6-8