Transdisciplinary Engineering Methods for Social Innovation of Industry 4.0
M. Peruzzini et al. (Eds.)

© 2018 The authors and IOS Press.
This article is published online with Open Access by IOS Press and distributed under the terms of the Creative Commons Attribution Non-Commercial License 4.0 (CC BY-NC 4.0).
doi:10.3233/978-1-61499-898-3-817

Group Technology: Genetic Algorithm Based on Greedy Constructive Structure and Refinement by k-Means Method Applied to Manufacturing Cell Formation Problems

Rogério Malta BRANCO[1] and Carlos Rodrigues ROCHA
Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul, Brazil

Abstract. The Group technology (GT) constitutes a manufacturing philosophy that exploits similarities in product design and product processes. The information attained from the process sheets are organized in an incidence matrix with machines and parts. The goal is to assign parts to families and machine to cells, which are designed to produce a given part family such that the number of voids and exceptional elements in cells are minimized. This article considers the problem of the manufacturing cell formation, of combinatorial nature and proposes a hybrid genetic algorithm (GA) with a greedy cosntrutive method for its solution, aiming the minimizing of the inter-cell movement and maximizing the use of the machines inside a cell. Basically, the GA generates sets of machine cells, and the constructor method is applied to those cells to better assign part families to them. The k-means algorithm is also applied to refine these formations. The performance of the proposed framework, considering the efficacy of grouping and a set of GT problems available in the literature, is presented and discussed.

Keywords. Group technology, machine-component cells, genetic algorithm, grouping efficacy

Introduction

The challenges of a globalized economy lead industries to define their strategic positions and, among others, to obtain good competitive advantages, crucial for their survival in the market. Also, as far as their productivity is concerned, companies already recognize the importance of production optimization, with special attention to layouts and their respective projects.

In this scenario, Group Technology (GT), a manufacturing philosophy based on the principle of identifying and grouping machines and parts by similarity, is important, given the advantages obtained in all the design and manufacturing stages [1].

The cellular manufacturing layout is a result of the GT applied in productive processes and includes several steps, among them the Production Flow Analysis (PFA). This results in a binary machine-part incidence matrix. Cells are the arrangement of

[1] Corresponding Author, Mail: rogerio.branco@riogrande.ifrs.edu.br .

these two types of elements, however they need to be optimized looking to minimize cross-cell flow and maximize intracellular motions.

Problems of machine clusters based on binary machine-part incidence arrays are considered NP-arduous because of their combinatorial nature.

In the literature related to the MCFPs there are a considerable number of methods applied in their solution, where some of the best known and used for quality comparison are: rank order clustering – ROC[2], modified rank order clustering - MODROC[3], Zodiac[4], Grafics[5], MST[6], GATSP[7], GA[8], similarity coefficient method- SLC[9], among others.

Specifically, in the case of heuristics applied to the topic, other works, such as the simulated annealing (SA) approach presented by Rao[10], which uses the grouping efficacy index to measure the effectiveness of the grouping. With the same objectives, also the proposals based on the GAs of Roy and Komma[11] and the particle swarm optimization algorithm (PSO) approach presented by Husseinzadeh Kashan et al.[12].

Recently, algorithms, modified to fit the structure of clustering problems, have attracted researchers. For example, James et al. [13] created cells applying their hybrid genetic algorithm, which combines a local search with a standard genetic clustering algorithm and concludes that a hybrid approach overcomes the standard clustering GA in terms of clustering effectiveness.

Immersed in this context, this work presents a hybrid GA aimed at the solution of MCFPs, applying k-means as a way to improve the individuals of a population. The objective function of the algorithm is the coefficient of efficacy, known to be a good index of performance of cellular clusters, and also because it is used in specific GT problems obtained from the literature which one wishes to compare results.

1. Machine-part cell formation problem

The problem of manufacturing cell formation (MCFP) in a binary matrix consists of the rearrangement of rows and columns to form families of parts and machine cells.

The main objective is to relate machine cells and parts families. In order to do so, it is necessary to use functions that guide the performance of these groupings, such as machine utilization[14], grouping efficiency [3] and grouping efficacy [15] , etc. Among the several measures of quality that guide the optimized formation of these groups, two are the most frequently used: efficiency and; the efficacy of grouping. Its popularity is due to the simplicity of application[14].

The mentioned grouping efficacy (μ) proposed by Kumar and Chandrasekaran [15] is adopted for two reasons: first because it overcomes the weaker discriminating power of grouping efficiency measure [3] by assigning equal weight for the number of voids and the number of exceptional elements; the second reason is because the results obtained in the works used as performance comparison apply this group quality indicator. This measure is defined as follows:

$$\mu = \frac{e - e_0}{e + e_v} \tag{1}$$

where e is the total number of operations (1's) in the given matrix, e_v is the number of

voids (0's in the diagonal groups), and e_o is the number of exceptional elements (1's out the diagonal groups).

Also, Paydar e Saidi-Mehrabad[14] and Goncalves and Resende [16] offer numerous other justifications for their adoption as, among others: incorporates both the with-in cell machine use and the inter-cell movement; generates block diagonal matrices which are interesting in practice; it is independent from the number of cells, etc.

Figure 1 shows an example a reordered binary machine-part incidence matrix. The green box indicates a void and a yellow, an exceptional element. A void means that, despite the machine and the part belong to the same group, this machine will not process the part. In other hand, an exceptional means that a intercellular movement will be necessary, since the part will be processed not exclusively inside de designed cell. Voids are not desirable, because they turn the cell more inefficient. The exceptional parts, in the same way, cause increasements in the processing time, among other complications.

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

Figure 1. Example of a two cell formation of a reordered incidence matrix.

Considering the example and the Eq. 1, it's possible to calculate the efficacy coefficient of the grouping, since the e, e_0 and e_v can be easily found:

$$\mu = \frac{18-2}{18+2} = 0{,}8 = 80\%$$

2. The proposed heuristic for the problem of cell formation

This paper presents an approach to the MCFP problem, involving genetic algorithm with a local search method to refine the solutions.

The GAs allow to reach a region close to the optimum with brevity. Also, achieving better results may require more time, so procedures are used to accelerate these convergences. A common technique is neighborhood search, where an GA solution is used as the starting point for another, faster and more efficient optimization solver for local search.

Aiming to minimize processing time and maximize quality of results, the k-means algorithm is aplied on a fully constructed chromosome in order to, by small exchanges, promote improvements in fitness, ie the efficacy of clusters.

2.1. The genetic algorithm with local search

Genetic algorithms are search algorithms based on the mechanisms of natural and genetic selection. They combine survival of the individual among other individuals, with chain structures created, initially at random.

The first work on this line was presented by Holland[17], aiming to replicate the processes used by the self-adaptive systems in a computational context. In fact, its objectives were to base a general theory of systems and robust adaptation, yet finding an excellent practical application, maximizing (or minimizing) mathematical functions. The GAs differ from the various heuristic methods because they present a group of distinct characteristics: operate on a set of points (population) and not from isolated points; operate in a space of coded solutions and not directly in the search space; they need as information only the value of an objective function (adaptability function, or fitness); use probabilistic transitions rather than deterministic rules [18].

In general terms, an initial population is generated (the chromosome of each individual is formed randomly) and the fitness value is calculated for each individual. Genetic operators are applied to probabilistically selected individuals, based on their fitness, and a new generation of individuals is created. However, the evolution of new generations is also driven by the insertion of pairs of chromosomes in the current population using crossover and mutation. Only the chromosomes that present better fitness will have more chances in the selection. This procedure will repeat itself until an end criterion is reached.

2.1.1. The chromosome: representation and decoding

The chromosomes denote feasible solutions to the problem and their length will be equal to the number of machines plus the number of parts presented in the MxN matrix.

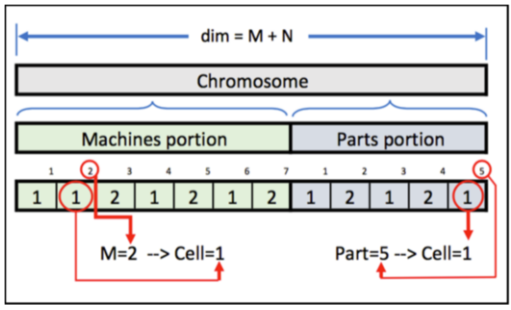

Figure 2. Example of a chromosome denoting the two cell formation presented in Figure 1.

Each gene in the chromosome string denotes a cell and, each locus, the machine (or part, depending on the portion under analysis) assigned. Thus, two information is contained in each locus (position) of the individual's string.

2.1.2. Genetic operators

As mentioned, genetic operators will be applied in the GA: cloning, crossover and mutation. Cloning a chromosome is just copying it to the next generation. In addition, the crossover determines the mechanism of combining two existing chromosomes and

creates two offsprings. These ones are selected by the roulette method [18], which gives greater chances to the fittest.

To combine them, a random binary mask is also generated with the intent to coordinate the offsprings formation process. A “0” in the mask means that the information shoud be taken from the f1. Otherwise, f2 will be the information giver.

At this point it is important to mention that, unlike the cloning operator, the others will only operate on the "machine" portion of the chromosome. The "pieces" portion is the result of the processing of the constructor algorithm, better seen later.

Already, combined with the crossover operator, the mutation refers to random changes in the genes to escape local maximums and guarantee access to any solution of the space. This operator acts on the new individuals generated, with a probabilistic rate of occurrence to each one, as can be seen in the Figure 3.

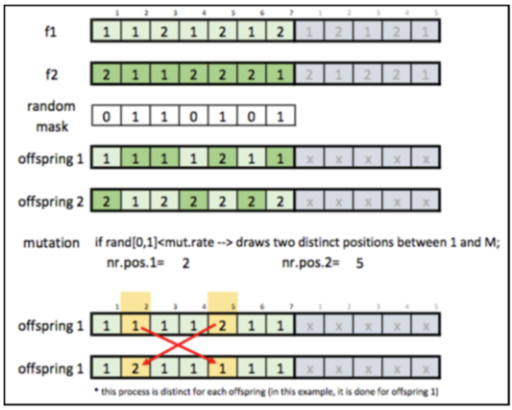

Figure 3. The crossover and mutaion operators.

2.2. The greedy construtive heuristic

It is observed that in both the random creation of individuals and in the application of the crossover and mutation operators, only the portion of the chromosome referring to the machines are processed (as can be seen in Figure 3).

As it is briefly shown in Figure 4, the constructor starts with the "pieces" portion of the chromosome already formed by some method (ie random, xover/mutation, etc.).

Algorithm: final chromosome constructor

Sm=matrix of similarity (machines);
I=individual with the machine portion already created;
k=predefined number of cells;
With I, do:
Create k-groups of machines (cells) assigning the most similar among them;
Create k-groups of pieces (families) designating each one, those with more frequent processing in each of the newly created cells; one piece can only be assigned to a single family;

Figure 4. An individual chromosome constructor algorithm.

To create a group of machines with this greedy heuristic, it is necessary to have the similarity matrix, created through the incidence matrix (better explained later). Of

these, the clustering starts doubling with the greatest similarities of the generated table. Cases where there are no links result in individual “islands” to be better grouped later.

In the second step, the similarities between the groups (and “islands”) are calculated, by the sum of the individual similarities of each member of a group with each member of the other. Those with higher total values come together and form larger groups. This occurs until the size k (predefined number of cells) is reached for the groups. at the end, we have the desired k-groups, with the greatest similarities among their elements.

Considering the formation of pieces, besides their matrix similarity, it is necessary to have the "machines" portion of the chromosome.

In short, the procedure evaluates the frequency of occurrence of the parts, according to incidence matrix, in the groups of machines already formed. Those whose appearance is more frequent to a particular group, are assigned to it. The others, make up a list of eligible parts designation according to the largest number of processing of this part within each of the cells, when competing for various machines. Having no more eligible pieces, the loop ends and the k-groups have been created.

2.3. The k-means procedure

The k-means is one of the most applied non-hierarchical methods, that is, they produce partitions of the set of objects with previous knowing the quantity of these.

It starts by defining k-centroids and, in successive iterations, groups the elements by criteria, such as least squares. The first centroids can be defined at random. Each iteration is recalculated as the average of the points belonging to its group. The iterations cease when there are no further changes in the centroids.

In this case, the lines of an incidence matrix can be seen as binary vectors, for the purpose of calculating similarity or distance between points, which in this case has dimension equal to the number of pieces (n). In the same way, when transposed, the binary matrix emphasizes the pieces and each vector-piece will have dimension equal to the number of machines. Eq. 2 shows the distance equation, where x_i and y_i are the vectors; i = {1,…n} is the index that points to each position of the vectors.

$$d = \sum_{i=1}^{n} \left\| x_i - y_i \right\| \tag{2}$$

The calculation of the coefficient of similarity between two vectors (lines or columns) consists in the identification of the coincidences of 1's between them. To do this, an AND logic is applied between the bits of the binary vectors. A non-zero bit in Sij indicates that the corresponding part in the machine-component chart should be processed on both machines i and j. The higher the result, the greater the similarity between the evaluated elements. In this way, an array can be formed with the values of similarities between each of the elements with each other: the similarity matrix. The Eq. 3 represents the similarity between vectors Ri and Rj:

$$S_{ij} = \left(R_i \text{ and } R_j \right) \tag{3}$$

The proposed framework starts this algorithm considering a fully generated chromosome, promoting modifications in the "machines" part.

Applying the principle of k-means and starting from the centroids of these groups, the goal is to perform movements without being, however costly, so the reason to do only one round of the algorithm. In tracing an internal structure to the genetic algorithm, one can consider these movements as local searches.

By calculating the centroids, it promotes exchanges based on the distances between the elements. After rebuilding the chromosome, the procedure on the "parts" partition is applied. The fully remodeled chromosome has calculated its adaptation, which, if better than the original chromosome, will replace it in the population.

3. Computational results

In order to compare the proposed framework with some others found in the literature, a program is developed in Python language to implement the algorithm. Also, the computer used in the tests is a notebook with i7 processor, model 7700HQ of 2.8GHz with 16GBytes of RAM. Although it was not the scope of the work, the implementation of the algorithms was done in Python, proved to be more than feasible for the construction of the necessary programming codes. In terms of performance, since it is an interpreted language, compared with another compiled one, it may perform poorly. Moreover, it is a computational language of easy development and vast material for support, besides being free.

The 24 problems chosen for the tests are provided by Gonçalves and Resende [16]. The aforementioned authors also compare their proposal with other works, also listed. In its totality, considering what is presented, this proposal exceeds the average values presented. In some cases, the maximum value of effectiveness found is briefly lower than that found in other studies, but most of them match or exceed them. The information regarding the problems studied is present in Table 1.

Table 1. Problems obtained from the literature for analysis.

Problems	Autor (source)	Size	Nr. cells
p01	King, Nakorchai (1982) [19]	5x7	2
p02	Waghodekar, Sahu (1984) [20]	5x7	2
p03	Seifoddini (1989) [21]	5x18	2
p04	Kusiak (1992) [22]	6x8	2
p05	Kusiak, Chow (1987) [23]	7x11	3
p06	Chandrasekharan, Rajagopalan (1989)[24]	8x20	3
p07	Chandrasekharan, Rajagopalan (1989) [24]	8x20	2
p08	McCormick et al. (1972) [25]	16x24	6
p09	Srinivasan et al. (1990) [26]	16x30	4
p10	King (1980) [2]	16x43	5
p11	Carrie (1973) [27]	18x24	6
p12	Mosier, Taube (1985) [28]	20x20	5
p13	Carrie (1973) [27]	20x35	4
p14	Chandrasekharan, Rajagopalan (1989) [24]	24x40	7
p15	Chandrasekharan, Rajagopalan (1989) [24]	24x40	7
p16	Chandrasekharan, Rajagopalan (1989) [24]	24x40	9
p17	Chandrasekharan, Rajagopalan (1989) [24]	24x40	9
p18	Chandrasekharan, Rajagopalan (1989) [24]	24x40	9
p19	McCormick et al. (1972) [14]	27x27	4
p20	Carrie (1973) [27]	28x46	9
p21	Kumar, Vannelli (1987) [29]	30x41	11
p22	Stanfel (1985) [21]	30x50	11
p23	McCormick et al. (1972)[14]	53x37	2
p24	Chandrasekharan, Rajagopalan (1989) [24]	100x40	10

The algorithm starts with the formation of population: 10% is intended for cloning of the best individuals, 10% for the formation of random individuals, and the remainder, designated for crossing and mutation. The population size is in 150 individuals, regardless of the size of the problem, as well as the mutation rate, by a value of 3%.

Table 2 contains the information obtained from Gonçalves and Resende [16] about results of the algorithms: Zodiac[4], Grafics[5], MST[6], GATSP[7], GA[8] and their own proposal, here denominate G&R.

Table 2. Comparisons of Percentage of Grouping Efficacy with different methods obtained from the literature for analysis.

	Zodiac	Grafics	MST	GATSP	GA	G&R
p01	73,68	73,68				73,68
p02	56,52	60,87			62,50	62,50
p03	77,36			77,36	77,36	79,59
p04	76,92			76,92	76,92	76,92
p05	39,13	53,12		46,88	50,00	53,13
p06	85,24	85,24	85,24	85,24	85,25	85,25
p07	58,33	58,13	58,72	58,33	55,91	58,72
p08	32,09	45,52	48,70			52,58
p09	67,83	67,83	67,83			67,83
p10	53,76	54,39	54,44	53,89		54,86
p11	41,84	48,91	44,20			54,46
p12	21,63	38,26		37,12	34,16	42,94
p13	75,14	75,14	75,14	75,28	66,30	76,22
p14	85,11	85,11	85,11	85,11		85,11
p15	73,51	73,51	73,51	73,03	73,03	73,51
p16	20,42	20,42	51,81	49,37	37,62	51,97
p17	18,23	44,51	44,72	44,67	34,76	47,06
p18	17,61	41,67	44,17	42,50	34,06	44,87
p19	52,14	41,37	51,00			54,27
p20	33,01	32,86	40,00			44,62
p21	33,46	55,43	55,29	53,80	40,96	58,48
p22	21,11	47,96	46,30	45,93	37,55	50,51
p23	52,21	52,21				56,42
p24	83,66	83,92	83,92	84,03	83,90	84,03

In all, for each problem, 30 rounds of the genetic algorithm are performed. With this data, statistics are generated and reported in Table 3. Also, comparative data between the proposed algorithm and the those obtained in the literature (Table 2) are also related.

Figure 5 presents the result of the diagonalization of the problem proposed by Waghodekar and Sahu [20]; one of the problems that had the result of the literature [5] surpassed by the proposed framework (11,3%), as presented in Table 3.

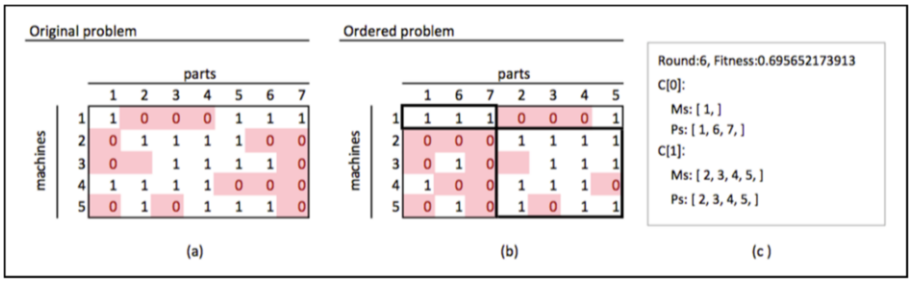

Figure 5. P02 original (a), ordered by the algorithm (b) and the information result from the program (c).

Table 3. Comparison of this approach and the literature.

	Approach results				Improvement: Table 2 vs. approach	
	Efficacy (μ)	Computational time (s)				
	Max	Min	Avg	Max	Avg	Max
p01	73,68	0,01	0,02	0,06	0,0%	**0,0%**
p02	69,57	0,02	0,03	0,05	14,8%	**11,3%**
p03	79,59	0,02	0,04	0,14	2,1%	**0,0%**
p04	76,92	0,02	0,03	0,07	0,0%	**0,0%**
p05	56,25	0,53	1,52	2,52	16,1%	**5,9%**
p06	85,25	0,02	0,04	0,00	0,0%	**0,0%**
p07	58,33	0,26	0,30	0,08	0,5%	-0,7%
p08	49,52	64,74	64,74	0,37	10,7%	-5,8%
p09	67,83	0,132	1,54	0,00	0,0%	**0,0%**
p10	54,60	15,28	61,47	64,74	0,6%	-0,5%
p11	53,10	9,25	9,25	8,12	12,1%	-2,5%
p12	39,23	125,14	125,14	107,65	12,7%	-8,6%
p13	76,14	0,48	0,51	125,14	3,1%	-0,1%
p14	85,11	0,19	0,23	0,00	0,0%	**0,0%**
p15	73,51	0,18	0,25	0,59	0,2%	**0,0%**
p16	49,02	1,75	8,92	0,00	27,0%	-5,7%
p17	42,02	13,12	13,12	0,33	7,8%	-10,7%
p18	39,57	1,710	5,54	0,47	5,6%	-11,8%
p19	53,54	20,10	39,81	24,92	7,7%	-1,4%
p20	37,80	131,08	131,08	13,12	0,5%	-15,3%
p21	59,35	184,74	184,74	14,99	19,7%	**1,5%**
p22	51,89	16,72	16,72	131,08	24,8%	**2,7%**
p23	56,72	12,61	22,20	184,74	5,8%	**0,5%**
p24	84,03	2,03	21,96	0,00	0,1%	**0,0%**

4. Final considerations

In this study, a hybrid genetic algorithm is proposed with the aim of maximizing grouping efficiency in cell manufacturing problems. The results were tabulated in Table 3 considering fixed parameters for 24 problems obtained in the literature and after, compared with 6 proposals known for their solution. The results surpassed in all the average values obtained by these 6 frameworks.

As can be seen in Table 3, comparing the results of the literature (Table 2), the algorithm obtains, for 7 (29%) problems values of grouping efficacy that are equal to the best ones. Also, it improves for 5 (20,8%). In 2 of them, the increasements are superior to 5%. Also, it is observed that all the average values obtained surpassed the means of results among the other applied methods. Also, for the 33% of the cases, the best solutions were already found in the first population.

Although the Python language performs less than the compiled languages, the time to obtain the results was satisfactory. Associated with the fact that k-means is applied to refine the results, however it is very computationally costly. Despite this, it allows to accelerate the convergence in many of the studied cases.

References

[1] S. A. Irani, S. Subramanian and Y. S. Allam, Introduction to Cellular Manufacturing Systems *Handb. Cell. Manuf. Syst.* , 2001, pp. 2–24.

[2] J. R. King, Machine-component grouping in production flow analysis: an approach using a rank order clustering algorithm, *Int. J. Prod. Res.*, Vol. 18, 1980, No. 2, pp. 213–232.
[3] M. P. Chandrasekharan and R. Rajagopalan, Modroc: An extension of rank order clustering for group technology, *Int. J. Prod. Res.*, Vol. 24, 1986, No. 5, pp. 1221–1233.
[4] M. P. Chandrasekharan and R. Rajagopalan, Zodiac—an algorithm for concurrent formation of part-families and machine-cells, *Int. J. Prod. Res.*, Vol. 25, 1987, No. 6, pp. 835–850.
[5] G. Srinivasan and T. T. Narendran, GRAFICS—a nonhierarchical clustering algorithm for group technology, *International J. Prod. Res.*, Vol. 293, 1991, pp. 463–478.
[6] G. Srinivasan, A clustering algorithm for machine cell formation in group technology using minimum spanning trees, *Int. J. Prod. Res.*, Vol. 32, 1994, pp. 2149–2158.
[7] C. H. Cheng, Y. P. Gupta, W. H. Lee and K. F. Wong, A TSP-based heuristic for forming machine groups and part families., *International J. Prod. Res.*, Vol. 36, 1991, No. 5, pp. 1325–1337.
[8] G. C. Onwubolu and M. Mutingi, A genetic algorithm approach to cellular manufacturing systems., *Comput. Ind. Eng.*, Vol. 39, 2001, No. 1–2, pp. 125–144.
[9] T. H. Wu, C. C. Chang and J. Y. Yeh, A hybrid heuristic algorithm adopting both boltzmann function and mutation operator for manufacturing cell formation problems, *Int. J. Prod. Econ.*, Vol. 120, 2009, p. 669-688.
[10] P. K. Rao, A multi stage heuristic for manufacturing cell formation., *Int J Res Eng Technol*, Vol. 1163, 2014, No. 45, pp. 2308–2321.
[11] N. Roy and V. R. Komma, Cellular manufacturing through composite part formation: a genetic algorithm approach., in *International Conference on Industrial Engineering and Operations Management*, 2014.
[12] A. Husseinzadeh Kashan, B. Karimi and A. Noktehdan, A novel discrete particle swarm optimization algorithm for the manufacturing cell formation problem, *Int. J. Adv. Manuf. Technol.*, Vol. 73, 2014, No. 9–12, pp. 1543–1556.
[13] T. L. James, E. C. Brown and K. B. Keeling, A hybrid grouping genetic algorithm for the cell formation problem, *Comput. Oper. Res.*, Vol. 34, 2007, Issue 7, pp. 2059-2079.
[14] M. M. Paydar and M. Saidi-Mehrabad, A hybrid genetic-variable neighborhood search algorithm for the cell formation problem based on grouping efficacy, *Comput. Oper. Res.*, Vol. 40, 2013, No. 4, pp. 980-990.
[15] C. Suresh Kumar and M. P. Chandrasekharan, Grouping efficacy: A quantitative criterion for goodness of block diagonal forms of binary matrices in group technology, *Int. J. Prod. Res.*, Vol. 28, 1990, No. 2, pp. 233–243.
[16] J. F. Gonçalves and M. G. C. Resende, An evolutionary algorithm for manufacturing cell formation, *Comput. Ind. Eng.*, Vol. 47, 2004, No. 2–3, pp. 247–273.
[17] J. H. Holland, Genetic Algorithms, *Sci. Am.*, Vol. 267, 1992, No. 1, pp. 66–72.
[18] H. P. L. Goldbarg and M. C. Luna, *Otimização combinatória e programação linear: modelos e algoritmos.*, 2ed ed., Elsevier Ltd., Rio de Janeiro, 2005.
[19] J. R. King and V Nakornchai, Machine-component group formation in group technology: review and extension, *International Journal of Production Research*,Vol. 20, 1982, Issue 2, pp. 117-133.
[20] P. H. Waghodekar and S. Sahu, Machine-component cell formation in group technology: Mace, *Int. J. Prod. Res.*, Vol. 22, 1984, No. 6, pp. 937–948.
[21] H. Seifoddini, Single linkage versus average linkage clustering in machine cells formation applications., *Comput. Ind. Eng.*, Vol. 16, 1989, No. 3, pp. 419–426.
[22] A. Kusiak and M. Cho, Similarity coefficient algorithms for solving the group technology problem, *Int. J. Prod. Res.*, Vol. 30, 1992, No. 11, pp. 2633–2646.
[23] A. Kusiak and W. S. Chow, Efficient solving of the group technology problem, *J. Manuf. Syst.*, Vol. 6, 1987, No. 2, pp. 117–124.
[24] M. P. Chandrasekharan and R. Rajagopalan, GROUPABILITY: An analysis of the properties of binary data matrices for group technology, *Int. J. Prod. Res.*, Vol. 27, 1989, No. 6, pp. 1035–1052.
[25] W.T. Mccormick, P.J. Schweitzer and T.W. White, Problem Decomposition and Data Reorganization by a Clustering Technique, *Oper. Res.*, Vol. 20, 1972, No. 5, pp. 993–1009.
[26] G. Srinlvasan, T.T. Narendran and B. Mahadevan, An assignment model for the part-families problem in group technology, *Int. J. Prod. Res.*, Vol. 281, 1990, No. 28, pp. 145–152.
[27] A.S. Carrie, Numerical taxonomy applied to group technology and plant layout, *Int. J. Prod. Res.*, Vol. 11, 1973, No. 4, pp. 399–416.
[28] C. Mosier and L. Taube, Weighted similarity measure heuristics for the group technology machine clustering problem, *Omega*, Vol. 13, 1985, Issue 6, pp. 577-579.
[29] K. R. Kumar and A. Vannelli, Strategic subcontracting for efficient disaggregated manufacturing, *Int. J. Prod. Res.*, Vol. 24, 1987, pp. 387–397.