MANUFACTURING CELL FORMATION USING SIMILARITY COEFFICIENTS AND A PARALLEL GENETIC TSP ALGORITHM: FORMULATION AND COMPARISON

Jaydeep Balakrishnan
Finance and Operations Management Area
Faculty of Management
University of Calgary
Calgary Alberta T2N 1N4
CANADA

Internet: balakris@ucalgary.ca

Prasanna D. Jog
Department of Computer Science
DePaul University
Chicago IL 60604
USA

Published in *Mathematical and Computer Modelling*, 21,12, 1995, 61-73.

MANUFACTURING CELL FORMATION USING SIMILARITY COEFFICIENTS AND A PARALLEL GENETIC TSP ALGORITHM: FORMULATION AND COMPARISON

Abstract

Many algorithms have been proposed to form manufacturing cells from component routings. However, many of these do not have the capability of solving large problems. We propose a procedure using similarity coefficients and a parallel genetic implementation of a TSP algorithm that is capable of solving large problems of up to 1000 parts and 1000 machines. In addition, we also compare our procedure with many existing procedures using nine well known problems from the literature.

The results show that the proposed procedure compares well with the existing procedures and should be useful to practitioners and researchers.

Keywords: Cellular Manufacturing, Travelling Salesman Problem, Parallel Genetic Algorithms, Group Technology, Similarity Coefficients.

MANUFACTURING CELL FORMATION USING SIMILARITY COEFFICIENTS AND A PARALLEL GENETIC TSP ALGORITHM: FORMULATION AND COMPARISON[1]

1.0 INTRODUCTION

The increasingly competitive manufacturing environment has forced managers to modify their approach towards manufacturing systems in order to be successful. One such approach or philosophy is called Group Technology (GT). Though the concept has existed for decades, it has only recently become popular in North America. GT is based on the principle of grouping similar parts into families and this can lead to economies throughout the manufacturing cycle [1]. These part families may be formed on the basis of either design characteristics or manufacturing characteristics. Design characteristics may include size, shape and function. Manufacturing characteristics include the type and sequence of operations required.

This paper discusses one aspect of forming part-families based on manufacturing characteristics. This aspect is called Cellular Manufacturing (CM) and it is one of the important parts of GT. CM concentrates on the formation of cells or groups of machines that process one or more part-families. Benefits of GT are many [1]. However, manufacturing cells can also cause a loss in the flexibility of the system because machines will now be dedicated to a few part-families instead of being available for more general purposes. In addition, considerable employee education may be required during and after GT implementation. In this paper, a procedure called SC-TSP is developed. It forms manufacturing cells effectively for even large problems.

[1] **This research has been supported by a Future Fund Fellowship from the Faculty of Management at the University of Calgary.**

2.0 BACKGROUND

Many methods of manufacturing cell (MC) formation have been developed. For an extensive review of these see [2]. A more recent review of the algorithms, measures of performance and widely used problems can be found in [3]. Some of the more important and recent research is briefly discussed here. Three of the more popular approaches are the Bond Energy Algorithm or BEA [4]; the Rank Order Clustering Algorithm or ROC [5]; and the ROC2 [6]. All of these algorithms identify machine groups and part groups simultaneously. More recently, Askin et al. [7] identify a Hamiltonian Path Heuristic (HPH) approach to machine grouping. They report results superior to the ROC2 approach. Wei and Gaither [8] use a 0-1 binary programming approach for an optimal solution. They used XMP [9] implemented on a Cray supercomputer to solve the 0-1 problems, the largest of which was a 41 part 30 machine problem with 224 variables and 350 constraints. This method is quite flexible since it can incorporate many different types of constraints. Cannon and Hoffman [10] discuss a parallel processing 0-1 algorithm which has the XMP code embedded in the procedure. A 201 variable, 134 constraint problem took 37 minutes of CPU time using VAX 2000 systems. Boctor [11] presents another 0-1 model for smaller problems. He uses simulated annealing to solve larger problems. Askin and Chiu [12] use heuristic graph partitioning in order to solve a 0-1 formulation of the GT problem. They report good solution times. Kaparthi and Suresh [13] report successful results using neural networks to solve a 10000 part, 100 machine problem. Vakharia et al. [14] compare two artificial intelligence approaches; simulated annealing and tabu search. The find that simulated annealing is better than tabu search in both solution quality and solution time. Tabu search was unable to solve a 325 part 25 machine problem in a reasonable amount of time. Flynn and Jacobs [15] compare cellular and process layouts using simulation. They report that neither of these layout types dominates the other. Shafer and Meredith [16] compare six different cell formation algorithms using real data and computer simulation. They found that no algorithm was best for all situations. Miltenburg and Zhang [17] compare nine well-known algorithms including the ROC2, BEA and algorithms with similarity coefficients using popular problems from the literature and an experimental data set. Based on their results, they suggest the ISNC algorithm of Chandrasekharan and Rajagopalan [18] as a good general approach for forming cells. Wemmerlov and Hyer [19] surveyed companies that had been using manufacturing cells in order to determine a variety of factors about group technology. They found that over one third of the companies had used formal algorithms and that many companies also performed manual analyses such as modifying part routings.

The SC-TSP procedure proposed here is similar to the ones described above. But it has

the capability of solving larger problems than many of the other algorithms.

3.0 FORMING MANUFACTURING CELLS FROM A PART-MACHINE MATRIX

Many of the algorithms in cell formation use a part-machine incidence matrix, an example of which is shown in Figure 1.

Insert Figure 1 about here

This matrix shows the relationship between the parts and the machines used to process them. For example, part 1 is processed by machines 1 and 7. Part 4 is processed by machines 1, 3 and 7. Each cell in the matrix with a '1' is called an element. An example of cells formed from the matrix in Figure 1 is shown in Figure 2.

Insert Figure 2 about here

The matrix is now in a block diagonal form. Three cells can be formed. Machines 2 and 4 form the first cell, machines 1 and 7 form a second cell, and machines 3, 5 and 6 form the third cell. All the parts except 4 can be processed entirely within a cell. The element of part 4 that does not fit completely into any manufacturing cell is called an exceptional element. The objective of many of the cell formation algorithms is to minimize these exceptional elements since they represent inter-cell movements.

4 USING SIMILARITY COEFFICIENTS AND THE CRAFT ALGORITHM IN CELL FORMATION: THE SC-TSP PROCEDURE

4.1 Determining Similarity Coefficients

Similarity coefficients (SCs) define relationships between pairs of machines or parts. The closer the relationship is, the higher the SC. The SCs are determined for both parts and machines from the part machine incidence matrix. The two SCs used here are the ones by McAuley [20] for machines and by Carrie [21] for parts. Vakharia and Wemmerlov [22] and Mosier [23] compare different similarity coefficients and report that McAuley's SC performed quite well. McAuley's SC for machines is as follows:

$$m_{kl} = \frac{n_{kl}}{x_k + y_l + n_{kl}}$$

m_{kl} *Similarity coefficient between machines k and l*

n_{kl} *Number of parts processed on both k and l*

x_k *Number of parts processed on machine k but not on l*

y_l *Number of parts processed on machine l but not on k*

The SC between machines 1 and 3, (m_{13}) in Figure 1 may be calculated using this method. Part 1 is processed on machine 1 only and parts 3, 5 and 6 are processed on machine 3 only. In addition, part 4 is processed on both these machines. So m_{13} is 1/5 or 0.20. An m_{kl} is determined for all possible pairs of machines.

Carrie's SC is very similar to that of McAuley's and is given by:

$$p_{ij} = \frac{n_{ij}}{x_i + y_j + n_{ij}}$$

p_{ij} *Similarity coefficient between parts i and j*

n_{ij} *Number of machines processing both i and j*

x_i *Number of machines processing part i but not part j*

y_j *Number of machines processing part j but not part i*

Based on Carrie's coefficients, the SC between parts 1 and 4 (p_{14}) in Figure 1 can be calculated as follows: machines 1 and 7 process both parts and machine 3 processes only part 4. p_{14} is therefore 2/3 or 0.67. A p_{ij} is computed for all possible pairs of parts.

4.2 Manufacturing Cell Formation using the TSP

A procedure using the Travelling Salesman Problem (TSP) Algorithm is discussed now. The TSP can be described as: Given N cities, if a salesman starting from his home city is to visit each city exactly once and then return home, find the order of visits (the tour) such that the total distance travelled is minimum. The NP-Completeness issues of the TSP has been discussed in Garey and Johnson [24]. Lawler et al. [25] is an excellent source for algorithms on the TSP. Though the TSP is NP-Complete, a number of excellent heuristic algorithms are available to solve it.

Insert Figure 3 about here

Consider the two pairs of cities A,B and C,D in Figure 3. The cities in each pair are close to each other, but the pairs themselves are far apart. The distance between two cities is called an edge. The arrows in the figure represent edges. It is clear that a tour in which the person starts from A and goes through cities D, B and C in that order, tour ADBCA shown in Figure 3a, would be an inefficient tour. Tour ABDCA (Figure 3b) would be a much shorter tour. In other words regardless of where one starts, it is better to visit all the cities in that cluster before visiting a city in another cluster. Shuttling between different clusters is inefficient. If A, B, C and D were machines (or parts) instead of cities and the distances surrogates for the similarity between machines (small distances indicating high similarity) then a good tour like ABDCA would indicate two machine groups. A and B would form one cell and C and D would form another cell. So applying the TSP using similarity coefficients can identify part or machine groups. An explanation of the exact algorithm used is explained in the next section.

4.3 The Parallel Genetic TSP Algorithm

Suppose we have a TSP, X, and we desire to minimize the tour length, T, we can use a genetic algorithm to solve such a problem optimally or near optimally. Genetic algorithms were invented by Holland [26] and differ from standard search algorithms in that the search is conducted using information of a population of tours in X instead of just one tour. This reduces the risk of being trapped in a local optimum when using heuristic algorithms.

A serious drawback of genetic algorithms is their inefficiency when implemented on a sequential machine. However, due to the algorithm's inherently parallel properties, they can be successfully implemented on machines with parallel processors resulting in reduced computation time. The algorithm described here, a parallel genetic algorithm by Jog [27], is implemented using fine grained parallelism. In this type of parallelism, each processor in the machine would contain one tour. In coarse grained parallelism, more than one tour would reside in a processor. The next section explains the genetic algorithm in detail assuming 64 processors.

Insert Figure 4 about here

Figure 4 shows the typical layout of a genetic algorithm. The initial population, P(0) consists of 64 tours generated randomly. A heuristic algorithm, Lin's 2-opt algorithm (28), is employed to reduce the tour length for each of the tours.

Figure 5 shows the 2-opt operator which randomly selects two edges from the tour, here **ab** and **cd** and replaces them with two edges **ad** and **cb**, if **ad+cb** < **ab+cd**. The distances are Euclidean. This process is repeated until a terminating conditions such as a limit on the number of iterations or a CPU time limit is encountered. In our implementation, 30 evaluations was set as the limit. As shown in Figure 4, the genetic algorithm then enters a loop where there are two steps which lead to the new population P(t+1): (1) **selection** and; (2) **recombination**. In the selection step, the 64 processors are divided into 16 groups of 4 processors each. In each group, the better than average tours are retained. Some of the poorer tours are also retained based on probabilistic selection. Retaining poor tours is useful in achieving better long run solutions. In each group, since we have not retained every tour and yet we seek to have the same number of tours, some tours may be repeated. However, two copies of the same tour will mate with different tours in the **recombination** step giving rise to different offspring. So having copies of the same tour does not diminish the effectiveness of the algorithm. The 64 processes are divided into groups in order to reduce computation time. If global selection is done using all 64 processors, then the processors that finish early lie idle waiting for the others to finish. The **selection** step resembles the

'survival of the fittest principle'.

Insert Figure 5 about here

Next, in the **recombination** step, each of 64 **selected** tours generates one offspring using **crossovers**. (Other recombination methods are also available. For a further review of such procedures, see Jog et al. [29]). A **crossover** operation produces an offspring from two parent tours. Figure 6 shows a Grefenstette crossover operation [30]. Let parent A be one of the **selected** tours in P(0). A second **selected** tour from P(0) is then randomly chosen and compared to the first one as shown in Figure 6. Let us start from city 1. Looking at parents A and B, we pick the shorter of the two outgoing edges from city 1 as the edge for the offspring. In this case they are equal. Then we look at the outgoing edges for city 2. Edge 2-3 in A is shorter than edge 2-6 in B. Hence we choose edge 2-3 as an edge for the offspring. This is repeated for each city. If the shorter edge were to create a cycle in the offspring, we would have picked the other edge. If both edges were to create a cycle, an edge that does not form a cycle in the offspring is selected randomly. The number of tours in P(t+1) is equal to that of P(t) ie. the population remains the same. However, the parents are now replaced by the offspring. Again, even if the offspring is worse than the parent, in the long run **crossovers** are effective. In **crossovers** also, the different processors can work in parallel to create new offspring thus reducing computation time. After the **crossovers** are performed ,the 64 offspring tour lengths are reduced using the 2-opt procedure. Finally, the algorithm loops back to the **selection** step. The genetic algorithm terminates when the run time limit expires.

Insert Figure 6 about here

4.4 Using the Similarity Coefficients in the TSP

The SCs have to be converted into distance measures in order to be used in the TSP. The higher the SC or relationship between two machines or parts, the smaller should be the distance between the parts or machines in the TSP. Since SCs vary between 0 and 1, this conversion can be accomplished by subtracting the SCs from 1 as follows:

$$d_{ij} = 1 - p_{ij}$$
$$d_{kl} = 1 - m_{kl}$$

where d_{ij} and d_{kl} represent the distances between pairs of parts and machines respectively and are symmetric. Two TSPs are solved. One of them is solved using the distances between parts and the other is solved using the distances between machines.

4.5 Using the TSP to obtain part and machine sequences

In the example problem, for the initial part and machine sequences shown in Figures 7a and 7b, we can calculate the similarity coefficients for the parts and machines. These coefficients are converted into distances as explained in the previous section. These distances are then used in the parallel genetic TSP algorithm. Since the genetic algorithm is sensitive to the initial tour, between fifteen and thirty replications are done for each part or machine sequence and the best five in tour length are chosen. The number of replications is greater for the larger problems. The five best tours for the parts and the five best tours for the machines can be combined to give 5x5 or 25 different part and machine combinations. The best combination from the twenty five for our example problem is shown in Figure 8. Visual analysis along with the two evaluation measures to be discussed later were used to determine the best combination. Combining the parts and machines form Figure 8 into a matrix and filling in the '1's will give us the matrix shown in Figure 2.

Insert Figure 7a about here

Insert Figure 7b about here

Insert Figure 8 about here

5 COMPARISONS WITH OTHER ALGORITHMS

5.1 Problem Set

In order to test the effectiveness of the SC-TSP procedure, nine problems from the literature were solved. Table 1 shows the different problems and their characteristics.

Insert Table 1 about here.

Problems 1, 3, 4, 5 and 6 were solved in [17] using nine different algorithms. In these problems, the SC-TSP procedure results are compared to the results in that paper. In addition, problems 2, 3 and 4 and 9 were solved in [7] using the HPH method. Since the reported results in that paper are better better than that of the ROC2 algorithm, the SC-TSP results from these problems are also compared to the HPH method results. Finally, problems 7 and 8 were solved to optimality in [8]. The optimal solutions to these two problems serve as bench-marks to evaluate the SC-TSP performance.

5.2 Performance Measures for Cell Formation Solutions

The results are presented in the form of part-machine incidence matrices which by itself cannot convey the quality of the solution. Therefore, a method is needed to compare these matrices using evaluation measures. Two measures used in [17] are used here. They are: (a) The Grouping Measure; and (b) The Bond Energy Measure. These measures were chosen because together they measure the within cell utilization, inter-cell movement, and the ability to cluster the '1's together. Thus they are comprehensive in their evaluation of cell formation. The first measure is considered to be the primary measure, while the other is considered to be a secondary measure.

a) The Grouping Measure

Let n_1 denote the number of '1's in the diagonal blocks (which form the cells) in the matrix. P_i and Q_i represent the number of parts and machines in each cell i respectively. Then

$$e_1 = \frac{n_1}{\sum_{i=1}^{K} P_i Q_i}$$

where K is the number of cells in the matrix and e_1 is an indicator of the within cell density of a cell. Higher values of e_1 indicate greater similarity between the parts or machines in this cell. Ideally, we would want the whole cell filled with '1's. Also

$$e_2 = 1 - \frac{n_1}{n_1 + n_0}$$

where n_0 indicates the number of '1's outside the cells or inter-cell transfers. Lower values of n_0 and thus lower values of e_2 indicate fewer inter-cell transfers and better solutions. The grouping efficiency, 'e' is then given by

$$e = e_1 - e_2 \qquad -1 \le e \le 1$$

In Figure 2 there are three manufacturing cells with four, four, and nine '1's in each cell respectively. The sum of these is 17 (n_1). The first two cells have two machines and two parts each and the last cell has three each of parts and machines. Summation of the products of the parts and machines is 17. Thus, e_1 is 17/17 or 1. The number of exceptional elements, n_0 is 1, thus, e_2 is equal to (1-(17/(17+1))) or 0.056 leading to a grouping efficiency (e) of 0.944.

b) The Bond Energy Measure

If algorithms form clusters, this implies that the '1's should be close to each other. This is the objective of the BEA. The strength of the clustering can be computed by 'b', the average bond energy measure of a matrix where

$$b = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N-1} a_{ij}a_{i(j+1)} + \sum_{i=1}^{M-1}\sum_{j=1}^{N} a_{ij}a_{(i+1)j}}{\sum_{i=1}^{M}\sum_{j=1}^{N} a_{ij}}$$

The notations are the same as those in Table 1. The more closely linked the '1's are, the higher and better 'b' will be. This measure is a normalized one in order to help in comparisons. In the example in Figure 2, 'b' is 1.17.

6.0 RESULTS

The results presented here are based on the visual analysis of the solutions provided by the SC-TSP procedure. Out of the twenty five candidate solutions, the one in which the cells could be most clearly identified was selected. Often, more than one solution had clearly identifiable cells in which case more than one solution is presented. The results are shown in Table 2 where the SC-TSP results are compared to the SC-Seed in [17], the HPH, and 0-1 programming. In [17], with respect to the five problems included here, the SC-Seed performed best with regard to 'e'. The cells from the SC-TSP solutions of the nine problems are shown in the Appendix. All the computer runs were made on a HP 9000/375 computer with a Motorola 68030 processor. The maximum solution time for one replication of any part or machine TSP problem was 0.27 CPU second.

In problems 1, 3 and 5, the SC-TSP grouping measure, 'e', is virtually identical to that of the SC-Seed algorithm. The SC-TSP bond energy, 'b' is better than any of the nine algorithms in all these three problems. The SC-TSP bond energy is even higher than that of the BEA whose objective is to maximize the bond energy.

In problem 2, since multiple solutions were generated, two different solutions were identified (Figures 9a and 9b). In the three cell solution, the SC-TSP grouping and bond energy measures are identical to that of the HPH method. The two-cell solution grouping measure is higher than that of

the three cell solution and was not obtained by the HPH method. This two cell solution has less inter-cell movement and may be preferable if machines cannot be duplicated, or if part routings cannot be easily modified, or if inter–cell movement is expensive. Thus it is important for the decision maker to have more than one solution for analysis. In problem 3, though the grouping measures are the same, the HPH method obtains slightly better bond energy measures than the SC-TSP procedure.

In problem 4, the SC-TSP grouping measure is comparable to that of HPH, ISNC or SC-Seed and better than or comparable to the other seven algorithms. The SC-TSP bond energy is better than any of the algorithms compared to it except the BEA.

Insert Table 2 about here.

Insert Figure 9a about here.

Insert Figure 9b about here.

In problem 6, two solutions were identified by SC-TSP. The one-cell solution has a grouping measure comparable to the SC-Seed method. The bond energy is bettered only by the BEA. The second SC-TSP solution has two cells. Though the grouping measure in this case is lower, it has a higher within cell utilization (e_1 in Table 2). If the exceptional elements can be reduced by routing modifications or machine duplication, this may prove to be a better solution in practice because it breaks a large group into two smaller groups and this is one of the objectives of CM.

In problems 7 and 8 the SC-TSP method was compared to the optimal solutions of the 0-1 programming formulation in [8] where constraints existed on the number of machines in a cell. The SC-TSP procedure imposed no such constraints. In problem 7, where the number of machines in a cell had to be twenty or fewer in [8], the SC-TSP obtains a four-cell solution, while the 0-1 procedure identifies a two-cell solution. The SC-TSP solution has a higher grouping and bond energy measures. In problem 8, both the SC-TSP and the 0-1 procedure identify two solutions from one final matrix. In the first solution in [8], the number of machines had to be ten or less and in the second it was constrained to be five or less. The 'e' values for both solutions are identical for both procedures. The SC-TSP obtains better bond energy measures. So, even when compared to an optimal procedure, the SC-TSP results are good. Thus, it should compare well with other artificial intelligence approaches such as simulated annealing and tabu search which provide near optimal results. The SC-TSP also used considerably less computation time than 0-1 programming. From the data on comparable problems, SC-TSP is also considerably faster than simulated annealing and tabu search

(based on the CPU times in [14]). However, 0-1 programming, simulated annealing and tabu search can incorporate different types of constraints and thus, are more flexible. In problem 9, both evaluation measures are comparable for the SC-TSP and HPH methods.

From published computation times, it appears that that SC-TSP is considerably faster than the nine algorithms in [17], the HPH method, and the parallel 0-1 programming in [10].

CONCLUSION

The tests conducted indicate that the SC-TSP procedure is effective for problems with different matrix densities. In all the test problems, it performed well on the grouping measure which is the primary measure of the quality of the solution. This good performance is also due to its effectiveness in increasing the bond energy of a matrix which is the secondary measure of the quality of the solution. This measure indicates the ability of an algorithm to rearrange random part-machine incidence matrices such that it becomes easy to identify cells. As reported in [19] , many companies performed manual analysis. The ability to form clusters will ensure that a decision maker has good solutions to perform the manual analysis of forming cells that suit the particular environment. Thus, performing well on the secondary measure is important. In addition, since multiple solutions are generated, the decision maker now has the flexibility of choosing from many solutions so that the best solution for a particular situation can be selected. For example, as seen in Table 2, the three-cell solution for Problem 2 has higher within cell utilization (e_1) than the two-cell solution, but has higher inter-cell movement (e_2) also. If parts can be rerouted or machines duplicated, the inter-cell movement can be reduced and the three-cell solution may be preferable. If part re-routing or machine duplication is expensive, or inter-cell movement will cause significant problems, the two-cell solution may be preferable. So it is important to have multiple solutions to choose from.

A major advantage of the TSP is its ability to solve large problems. Tests have shown that the parallel genetic TSP algorithm can solve a 1000 city problem in about 2700 CPU seconds on a BBN Butterfly machine. Thus, this procedure can be used to form manufacturing cells in large job shops.

In summary, the effectiveness of the SC-TSP approach should make it useful for practicing managers and researchers alike. Further research could involve using an expert system along with this procedure for analyses such as selecting the best solution from a group of 25 solutions, modifying part routings, duplicating machines or subcontracting.

Appendix
Cells Created by SC-TSP

PROBLEM		CELL				
		1	**2**	**3**	**4**	**5**
1	**Parts**	8 5 2	9 6 3 4	1 7 10		
	Machine	8 5 3 15 13	7 11 10 12 2	4 6 9 14 1		
2	**Parts**	19 18 12 1	10 17 16 9	3 2 6 13 7 4 5 15 14 8		
	Machine	2 1 9 4	8 6 3 5	12 11 7 10		
	Parts	12 18 19 11 9 17 16 10	15 14 5 4 2 7 13 8 6 3			
	Machine	7 10 11 12	2 5 3 6 8 4 9 1			
3	**Parts**	7 18 4 20 6 3	10 12 15 1 5	9 17 11 14 13 19 8 2 16		
	Machine	2 8 4 7	5 6	1 3		
4	**Parts**	7 4 18 10 40 32 42 37 2 38 29	16 29 5 21 19 43 33 14 23 8 15 41 9	20 3 24 27 4 22 30 23 14 19 41 21 5	25 26 31 39 13 1 12	38 35 6 17 34
	Machine	1 16 9 2 6	8 4 5 15	13 11 12	10 7	14 3
5	**Parts**	30 28 32 21 9 11 4 6	16 14 8 19 26 22 7	27 2 13 24 12 10 18 31	3 1 15 17 5 29 25 23 20	
	Machine	19 11 16 15 12	20 9 10 6 5	4 13 18 2 14	3 17 8 7 1	
6	**Parts**	6 11 12 20 7 5 8 13 17 19 16	2 1 10 15 14 3 18 9 4			
	Machine	7 2 6 1	4 8 5 3			
7	**Parts**	27 16 34 36 17 5 37 4 8 29 9 14	22 21 3 1 30 13 38	41 33 10 32 31 39 12 40 23 11 2 20 19 18	6 24 25 7 35 28 15 26	
	Machine	15 5 26 7 17 18 8 28	13 24 20 30 19 29 9	11 1 2 21 22 3 23 10 12	27 4 16 6 14 25	
8	**Parts**	19 20 17 2 1 23	7 6 18 8	24 3 4 21	11 14 22 16 10 12 9 15 5 13	
	Machine	4 5 7	13 12 1	10 11 3 2	14 9 8 6	
	Parts	19 20 17 23 7 6 18 8 1 2	4 21 11 14 22 24 16 10 12 9 15 5 3 13			
	Machine	4 5 7 13 12 1	10 11 3 2 14 9 8 6			
9	**Part**	24 3 4 21	11 13 5 12 15 9 14 10 16 22	18 8 7 6	19 23 17 20 1 2	
	Machine	1 3 2	10 9 8	13 14 11 12	6 7 5 4	

REFERENCES

1 N.C. Suresh and J.R. Meredith, Achieving factory automation through group technology principles, **Journal of Operations Management**, 5, 2, 151-167 (1985).

2 U. Wemmerlov and N.L. Hyer, Procedures for the part family/machine group identification problem in cellular manufacture, **Journal of Operations Management**, 6, 2, 125-148 (1986).

3 C-H. Chu, Cluster Analysis in Manufacturing Cellular Formation, **OMEGA - The International Journal of Management Science**, 17, 3, 289-295 (1989).

4 W.T. McCormick Jr., P.J. Schweitzer and T.W. White, Problem decomposition and data reorganization by a clustering technique, **Operations Research**, 20, 5, 993-1009 (1972).

5 J.R. King, Machine component groupings in production flow analysis: an approach using rank order clustering algorithm, **International Journal of Production Research**, 18, 2, 213-232 (1980).

6 J.R King and V. Nakornchai, Machine-component group formation in group technology: review and extension, **International Journal of Production Research**, 13, 6, 657-579 (1982).

7 R.G. Askin, S.H. Cresswell, J.B. Goldberg and A.J. Vakharia, A Hamiltonian path approach to reordering the part-machine matrix for cellular manufacturing, **International Journal of Production Research**, 29, 6, 1081-1100 (1991).

8 J.C. Wei, and N. Gaither, An optimal model for cell formation decisions, **Decision Sciences**, 21, 2, 416-433 (1990).

9 R.E. Marsten, Users Manual for ZOOM/XMP. Tucson AZ: University of Arizona, 1987.

10 T.L Cannon and K.L. Hoffman, Large-scale 0-1 linear programming on distributed workstations, **Annals of Operations Research**, 22, 181-217 (1990).

11 F. Boctor, A linear formulation of the machine-part cell formation problem, **International Journal of Production Research**, 29, 2, 343-356 (1991).

12 R.G. Askin, and K.S. Chiu, A graph partitioning procedure for machine assignment and cell formation in group technology, **International Journal of Production Research**, 28,8, 1555-1572 (1990).

13 S. Kaparthi and N.C. Suresh, Machine- component cell formation in group technology: A neural network approach", **International Journal of Production Research**, 30, 6, 1353-1368 (1992).

14 A.J. Vakharia, Y.L. Chang, and H.M. Selim, Cell formation in group technology: a combinatorial search approach, **Proceedings of the 1992 Annual Meeting of the Decision Sciences Institute**, 1310-1312 (1992).

15 B.B. Flynn, and F.R. Jacobs, An experimental comparison of cellular (group technology)

layout with process layout, **Decision Sciences**", 18, 562-581 (1987).

16 M.S. Shafer, and J.R. Meredith, A comparison of selected manufacturing cell formation techniques, **International Journal of Production Research**, 28, 4, 661-673 (1990).

17 J. Miltenburg and W. Zhang, A comparative evaluation of nine well-known algorithms for solving the cell formation problem in group technology, **Journal of Operations Management**, 10, 1, 44-69 (1991).

18 M.P Chandrasekharan and R. Rajagopalan, An ideal seed non-hierarchical clustering algorithm for cellular manufacturing, **International Journal of Production Research**, 24, 2, 451-464 (1986).

19 U. Wemmerlov and N.L. Hyer, Cellular manufacturing in the U.S. industry: a survey of users, **International Journal of Production Research**, 27, 9, 1511-1530 (1989).

20 J. McAuley, Machine grouping for efficient production, **Production Engineer**, February, 53-57 (1972).

21 A. Carrie, Numerical taxonomy applied to group technology and plant layout, **International Journal of Production Research**, 11, 4, 399-416 (1973).

22 A.J. Vakharia and U. Wemmerlov, A new similarity index and clustering methodology for formation of manufacturing cells, **Proceedings of the 1988 Annual Meeting of the Decision Sciences Institute**, 1075-1077 (1988).

23 C.T. Mosier, An experiment investigating the application of clustering procedures and similarity coefficients to the GT machine cell formation problem, **International Journal of Production Research**, 27, 10, 1811-1835 (1989).

24 M. Garey and D.S. Johnson, **Computers and Intractability: A Guide to the Theory of NP-Completeness**, San Francisco: Freeman, 1979.

25 E.L. Lawler, J.K. Lenstra, H.G. Rinnooy Kan and D.B. Shmoys, **The Travelling Salesman Problem: A Guided Tour of Combinatorial Optimization**, New York: Wiley, 1985.

26 J.M Holland, **Adaption in the Natural and Artificial System**, University of Michigan Press, Ann Arbor, MI, 1975.

27 P.D. Jog, Parallelization of probabilistic sequential search algorithms, unpublished doctoral dissertation, Indiana University, Bloomington IN., 1989.

28 S. Lin, Computer solutions to the travelling salesman problem, **Bell Systems Technical Journal**, 44, 2245-2269 (1965).

29 P. Jog, J.Y. Suh and D. Van Gucht, Parallel genetic algorithms applied to the travelling salesman problem, **SIAM Journal of Optimization**, 1, 4, 515-529 (1991).

30 J.G. Grefenstette, R. Gopal, B.J. Rosmaita and D. Van Gucht, Genetic algorithms for the travelling salesman problem, **Proceedings of the International Conference on Genetic Algorithms",** J.J Grefenstette ed., 160-168 (1985).

31 H.M Chan and D.A. Milner, Direct clustering algorithm for group formation in cellular manufacture, **Journal of Manufacturing Systems**, 1, 1, 65-74 (1982).

32 J. De Witte, The use of similarity coefficients in production flow analysis, **International Journal of Production Research**, 18, 4, 503-514 (1980).

33 J.L. Burbidge, **The Introduction of Group Technology**, New York: Wiley, 1975.

34 M.P. Chandrasekharan and R. Rajagopalan, MODROC: An extension of rank order clustering for group technology, **International Journal of Production Research**, 25, 5, 1221-1233 (1986).

35 K.R. Kumar and A. Vanelli, Strategic subcontracting for efficient disaggregated manufacturing, **International Journal of Production Research**, 25, 12, 1715-1728 (1987).

36 L.E. Stanfel, Machine clustering for economic clustering, **Engineering Costs and Production Economics**, 9, 73-81 (1985).

Machine	Parts						
	1	2	3	4	5	6	7
1	1			1			
2		1					1
3			1	1	1	1	
4		1					1
5			1		1	1	
6			1		1	1	
7	1			1			

A Part-Machine Incidence Matrix

Figure 1

Machine	Parts						
	2	7	4	1	6	3	5
2	1	1					
4	1	1					
1			1	1			
7			1	1			
3			1		1	1	1
6					1	1	1
5					1	1	1

Manufacturing Cells

Figure 2

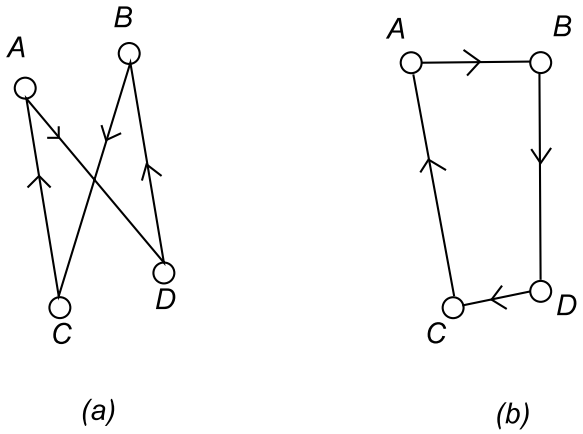

Figure 3

Relevance of the TSP in Cellular Manufacturing

```
P(t) denotes the population at time t.
t ← 0;
initialize P(t);
use 2-opt on P(t);
while (termination condition is not satisfied)
{
        t ← t+1;
        select P(t);
        recombine P(t);
        use 2-opt on P(t);
}
```

Figure 4

Layout of the genetic algorithm

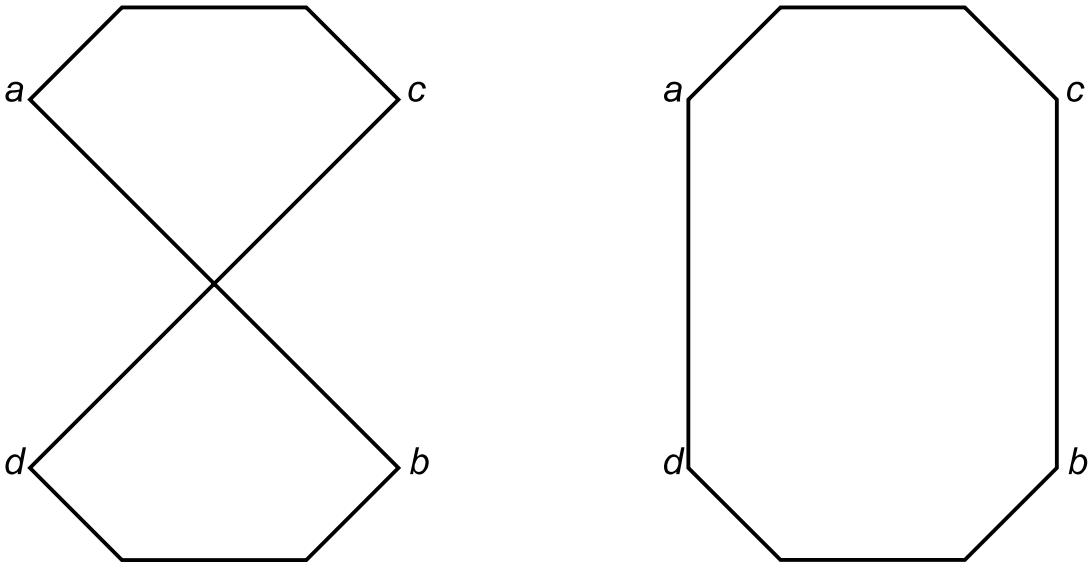

Figure 5

Illustration of a 2–opt operation

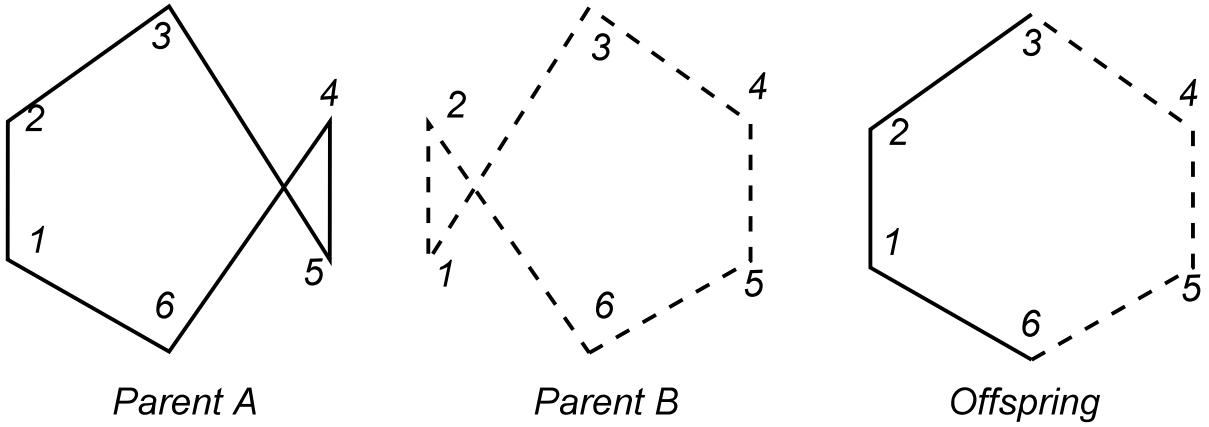

Figure 6

The Grefenstette crossover operator

1	2	3	4	5	6	7

Figure 7a

Initial part layout

1
2
3
4
5
6
7

Figure 7b

Initial machine layout

2	7	4	1	6	3	5

2
4
1
7
3
6
5

Figure 8

Final part and machine layout

MACHINES	PARTS																		
	1	19	18	12	11	9	10	17	16	15	14	5	4	2	7	13	6	8	3
2		1				1		1	1										
1						1	1	1	1		1								
9		1	1		1	1	1	1		1	1								
4		1	1	1	1	1	1	1	1										
8		1	1	1	1	1		1	1	1									
6	1	1	1	1	1				1	1	1	1							
3		1	1	1															
5		1	1																
12	1															1	1	1	1
11														1	1	1	1	1	1
7							1	1	1	1	1	1	1		1	1	1	1	
10										1	1	1	1	1	1	1			

Figure 9a
Rearranged matrix for Problem 2 with 3 cells

MACHINES	PARTS																		
	10	9	17	16	19	18	12	11	15	14	5	4	2	7	13	8	6	3
7	1		1	1					1	1	1	1		1	1	1	1	
10									1	1	1	1	1	1	1			
11													1	1	1	1	1	1
12															1	1	1	1
2			1	1	1													
5					1	1												
3					1	1	1											
6				1	1	1	1	1	1	1	1							
8		1	1	1	1	1	1	1	1									
4	1	1	1	1	1	1	1	1										
9	1	1	1		1	1		1	1	1								
1	1	1	1	1	1					1								

Figure 9b
Rearranged matrix for Problem 2 with 2 cells

Number	Problem	Parts (N)	Machines (M)	Matrix Density[1]
1	Chan &Milner [31]	10	15	0.31
2	De Witte [32]	19	12	0.33
3	Chandrasekharan and Rajagopalan [18]	20	8	0.38
4	Burbidge [33]	43	16	0.18
5	Carrie [21]	35	20	0.19
6	Chandrasekharan and Rajagopalan [34]	20	8	0.57
7	Kumar and Vanelli [35]	41	30	0.105
8	King [5]	24	14	0.175
9	Stanfel [36]	24	14	0.18

[1] The matrix density is given by $\frac{\sum_{i}^{M}\sum_{j}^{N} a_{ij}}{MxN}$ where $a_{ij} = (0,1)$, one element in the part machine matrix.

Test Problems

Table 1

	SC-TSP					SC-Seed		HPH Method		0-1 Programming	
	No. of cells	e_1	e_2	e	b	e	b	e	b	e	b
1	3	0.92	0.0	0.92	1.41	0.93	1.37	–	–	–	–
2	2	0.57	0.15	0.42	1.32	–	–	–	–	–	–
	3	0.68	0.31	0.37	1.29	–	–	0.37	1.29	–	–
3	4	1	0.15	0.85	1.34	0.85	1.31	0.85	1.38	–	–
4	5	0.65	0.21	0.44	1.21	0.45	1.02	0.42	1.11	–	–
5	4	0.79	0.03	0.76	1.52	0.76	1.40	–	–	–	–
6	1	0.57	0	0.57	1.30	0.57	1.21	–	–	–	–
	2	0.79	0.31	0.48	1.30	–	–	–	–	–	–
7	4	0.37	0.06	0.31	1.16	–	–	–	–	0.16[1]	0.91
8	2	0.33	0	0.33	1.19	–	–	–	–	0.33	1.08
	4	0.66	0.03	0.63	1.19	–	–	–	–	0.66	1.08
9	4	0.69	0.03	0.66	1.2			0.66	1.21		

NOTE: [1] Two cell solution

Results

Table 2