GENETIC ALGORITHM APPROACH TO MODELLING FRACTAL MANUFACTURING LAYOUT

Julian Aririguzo C[(a)] and Sameh M Saad [(b)]

[(a)(b)]Faculty of Arts, Computing, Engineering and Sciences
Sheffield Hallam University
Sheffield, UK

[(a)]J.Aririguzo@shu.ac.uk [(b)]S.Saad@shu.ac.uk

ABSTRACT

Fractal Manufacturing System (FrMS) basically structurally builds up from units called 'fractals' or fractal objects which are independent entities and contain essential features and congenital attributes of the entire manufacturing configuration. They can self-adapt quickly to dynamic changes in an unpredictable manufacturing environment. They are also self regulating and fall under organizational work groups, each within its own area of competence. An optimal shop floor design and implementation is key and an integral part of achieving a successful FrMS. and is concerned with issues of shop floor planning, arrangement and function layout. The fractal shop floor layout develops from individual cells and is conceptually capable of producing a variety of products with minimal reconfiguration. Keen attention is paid to determination of capacity level, cell composition and flow distances of products. In this paper, Genetic Algorithm (GA) is adopted to efficiently and effectively design flexible FrMS shop floor layout, needed in agile manufacturing system to cope with new and dynamic manufacturing environments that need to adapt to changing products and technologies. Its stochastic search algorithm is used in simulating natural evolutionary process techniques, which in turn solves the many FrMS combinatorial optimization problems. The design implementation is done using MATLAB. The end result interestingly is a fault tolerant structure that is better suited to survive and stand the pressure for lead time reduction and inventories, product customization and challenges of a dynamic and unpredictable operational environment.

Keywords: Fractal manufacturing system, Manufacturing layout design, Genetic algorithm.

1. INTRODUCTION

The conceptual fractal shop floor builds up from individual cells called fractals and is capable of producing a variety of products with minimal reconfiguration (Venkatadri *et al*. 1997; Montreuil *et al.*, 1999). This is due mainly to their ability to self-adapt quickly to dynamic changes in an unpredictable manufacturing environment. The fractal layout is an extension of the cellular layout (Askin *et al*. (1999) and in fact, each fractal cell is a multifunctional mini shop (Venkatadri *et al*., 1997) since it could produce most of the product types routed to it and have layout specification that produce varied products. This decentralized production layout allows for flexible mass customization. However, there are many challenges posed by the design and implementation of this strategy. A design and simulation of the model of shop floor layout for FrMS is presented in this paper paying attention to determination of capacity level and cell composition using genetic algorithm approach. The procedure is based on an iterative algorithm, implemented using MATLAB and used to calculate material travelling distances for each fractal cell and this continuously optimizes the layout, flow assignment and improves the overall performance of these parameters to create maximum space utilization. The rest of the paper is organized as follows; section two briefly looked at the general fractal manufacturing layout, section three made an overview of the GA and MATLAB and sets the scene for the application of the GA to the layout design and experimentation. Section four discusses the essentials of the proposed fractal manufacturing layout. Section five implements the GA approach, while section six discusses output results and the paper is finally concluded in section seven.

2. FRACTAL MANUFACTURING LAYOUT

The fractal workstation layout is created to minimize the capacity requirements and material travelling distances (Saad and Lassila 2004). The layout design concerns the arrangement of physical production resources within the production facility (Chase and Aquilano 1992) and the planning of which involves the determination and allocation of the available space to a given number of resources (Azadivar and Wang 2000) and emphasizes minimization of flow distances in order to improve product flow and general layout performance (Montreuil *et al*., 1999). This involves various aggregate steps; capacity planning, fractal cell creation, flow assignment and cell/ global layout (Venkatadri *et al*.1997). These processes can significantly affect the efficiency of the planned manufacturing system in terms of shop floor control, equipment utilisation, materials handling, materials management, and worker productivity (Co and Araar 1988). The manner and nature of the flow of materials through the facility is of crucial importance and this includes the flow rate, throughput time of products and the routes taken by the products (Wild 1993).

2.1. Capacity Planning

The consideration of the resource requirements, demand, capacity, work methods, handling and movement, departmental area requirements, and shape and location restrictions are all issues of capacity planning (Wild 1993, Gau and Meller 1999). The general and perhaps most important objectives are how to minimize the physical movement and handling of materials, maximize the capacity utilisation (Wild 1993) and ensure a smooth work flow (Chase and Aquilano 1992) in accordance with the system plan. Other very important issues include; systems design, machine reliability, parts scheduling, etc. These are all issues involved in the capacity planning process. It is worth mentioning that the capacity planning task requires optimal value of input data to satisfy product demand, minimize investment and operations cost and go into production within the pre-specified production time, though cost of material transfer could be traded-off against initial investment cost (Montreuil *et al*., 1999). This study first determines the required capacity levels for each machine type and the number and composition of fractal cells. Then an iterative algorithm continuously optimizes the layout and flow assignment according to the performance of the system.

2.2. Fractal Cell Creation

The fractal layout system uses cells to group machines together and to control and limit product routings. The number of fractal cells and workstation composition of each cell is significant in the overall manufacturing system. Its layout can be seen as an extension of a cellular layout (Askin *et al*. 1999) due to the structure of the shop floor, and fractal cells are multifunctional and are able to process most of the product types routed into the system. Each cell needs to contain exactly one replicate of workstation type. They also share workstations, but each cell has equal compositions. These identical cells are flexible and standardized. They can respond well to short term changes, uncertainties and unpredictable incidents or events such as machine breakdown, product mix, and transfer devices going offline (Montreuil *et al*., 1999).

2.3. Flow Assignment

The fractal cell, a set of neighboring workstations is the basic unit in the fractal layout system (Venkatadri *et al*. 1997 & Montreuil *et al*. 1999). The system flexibility is believed to increase because all fractal cells have roughly the same composition of machines and are capable of processing most of the products routed to them. It helps to alleviate flow congestion of products and improve flow efficiency. The flow score is measured and analyzed in order to estimate the frequency and distance travelled. The satisfactory estimation of flow around the actual workstations is also of significance in the layout design. Flow assignment involves the decision of getting the products processed through particular machines on the job shop (Askin *et al*.1996 & 1999). The assignment of products to flow paths minimizes travel distance if there are several products with specified machine type routing to be processed (Venkatadri *et al*., 1997). The objective of the flow assignment is to create a workstation layout that minimizes the capacity requirements and material travelling distances for a particular product. The flow assignment experiment and capacity analysis can be used to improve the layout repeatedly until a satisfactory layout is generated (Montreuil *et al*., 1999). The results indicate that unrestricted product flows offer the best flow scores in a fractal layout.

3. OVERVIEW OF GA AND MATLAB

Many combinatorial optimization problems in manufacturing systems are very complex and can not be solved using conventional optimization techniques (Kamrani and Gonzalez 2003). Most fractal manufacturing layout problems are dynamic - they change with time, and the system is expected to self-adapt to unpredictable changes and uncertainties. To deal with such problems efficiently and effectively, different fault tolerant structures and adaptable methods are required in solving problems in these dynamic operational environments.

3.1. GA procedure

GA is an evolutionary algorithm, a simulation of natural evolutionary process technique that has been adopted for this study (Azadivar and Wang 2000). The GA approach is a powerful and broadly applicable stochastic search technique used to find approximate solutions in optimization problems (Holland 1975). Just like evolutionary algorithms, it allows systems to self-adapt to make up for unpredictable changes in the operational environment. It would take into account a wider range of possible solutions and further increase the probability of finding optimal solution by continuously iterating and optimizing the design of the fractal layout and flow assignment according to the performance of these parameters.

3.1.1. Genetic Operators

Crossover and mutation are the two genetic operators that are applied probabilistically to create a new population of individual strings (Rajasekharan, 1998). Crossover is an important operation performed by GA for solving combinatorial optimization problem. Two of the individual strings in the initial population are selected randomly as two parents. A cut point is randomly chosen within the parent strings (Kamrani and Gonzalez 2003).

3.1.2. Crossover

Crossover operation exchanges cross sections of the parents in order to form two offspring. As shown in (Figure 1), the two off-springs form new individual strings generated by combining the “head” of the first parent string with the “tail” of the second parent string and vice versa (Rajasekharan, 1998). The essential characteristic of crossover is the crossover rate (CR) which is the ratio of number of off-springs produced in each generation to the population size. A higher CR allows deeper exploration of solution space and increases the chance of achieving accurate optimal results. If the CR is too high, it results in wastage of computational time (Kamrani and Gonzalez 2003).

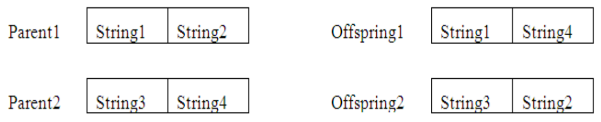


Figure 1: Crossover (Al-Sultan *et al.*, 1997)

Due to the unique hierarchical chromosome scheme used, a one-point crossover is used as in (Xiaodan Wu *et al.*, 2007). A cut point is randomly selected over the whole chromosome as shown in (Figure 2). Parent1 and Parent2 are the chromosome pair selected for the crossover operation. The “head” of Parent1 is replaced by “tail” of Parent2. Then Child1 is generated. On the other hand, the “tail” of the Parent1 replaces the “head” of the Parent2, and Child2 is then created.

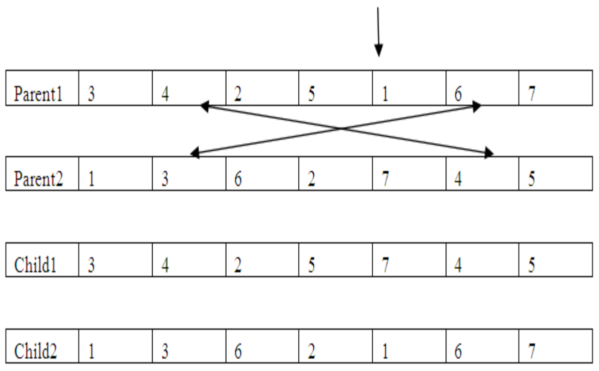


Figure 2: Numerical illustration of Crossover (Xiaodan Wu *et al.*, 2007)

3.1.3. Mutation

Mutation operation produces spontaneous random changes in certain chromosomes. Mutation play two roles that involve either replacing the genes lost from the population during the selection process, or providing the genes that were not present in the initial population (Kamrani and Gonzalez 2003). Mutation is designed to prevent premature convergence and to explore a new solution space (Xiaodan Wu *et al*. 2007). But the mutation operation alters and mutates one or more genes within the chromosomes of an individual rather than across a pair of chromosomes. There are two kinds of mutation proposed by (Xiaodan Wu *et al.* 2007), which are group mutation (Figure 3) and inverting mutation (Figure 4). Group mutation is for exchanging genes of the same group for the same layer at same time while inverting mutation involves exchanging the genes from the randomly chosen loci of the parent. Both genes are chosen randomly for the operation of mutation. From a theoretical perspective, if the length of the chromosome for inverting mutation is long, the chances of finding the optimal solution in the near-optimal area is low. However, the group mutation can help to enhance the GA’s ability of exploiting and converging rapidly to a promising region (Xiaodan Wu *et al.*, 2007). (Al-Sultan and Fedjki 1997) illustrated in (Figure 3) that group inverting mutation begins with a selection of a parent, and randomly dividing into two strings. The two strings are then exchanged to get a new offspring. Group inverting mutation involves two steps - a random cut of the selected parent is generated and the two chosen strings are then exchanged to obtain a new offspring.

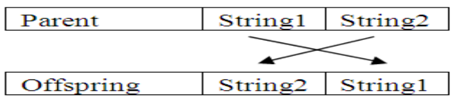


Figure 3: Group Mutation (Al-Sultan and Fedjki 1997)

There is yet another kind of inverting mutation by (Hicks 2006), that involves the selection of the two points randomly and then the genes between those points are placed in reverse order. This inverting mutation is shown in (Figure 4). The other genes in other positions are also copied directly from the parent to the child. In an inserting mutation, a gene is selected at random. The gene is taken off from the chromosome and then inserted back in a random position (Parames, 2001).

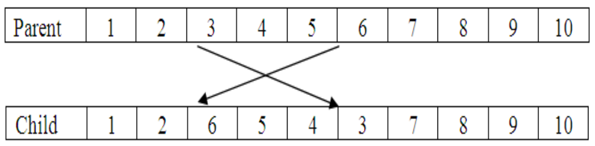


Figure 4: Inverting Mutation (Azadivar and Wang 2000)

3.1.4. Stopping Criteria

Two stopping conditions are employed to stop the GA from iterating continuously (Parames, 2001). First, if the number of iterations exceeds the predefined fitness value, GA would stop the operation immediately. The other stopping condition is that the value of the objective function does not change within the expected number of iterations. Once the algorithm has completed the given number of generations, the best value of the objective function is obtained. At that moment, GA would be terminated and display the layout configuration associated with the chromosome with the highest fitness value.

3.2. MATLAB R2008a

MATLAB is a high-level computer language for scientific computing and data visualization built around an interactive programming environment (Kiusalaas 2005). It integrates computing, visualization and programming in one user-friendly environment. In the course of this study, MATLAB R2008a is used to develop, implement, customize and create a user-friendly graphic interface (Kiusalaas 2005) for our fractal layout design. Its high-performance language essentially expresses the problems and solutions in mathematical notation. The model is then designed to fetch input of pair-wise comparison data of different criterion and alternatives and process these data to an output of optimum score of the alternatives.

A set of general procedures are employed in the design of the fractal shop floor layout. There are several phases to the procedure;

1. Design and simulate the model of FrMS shop floor layout using MATLAB R2008a, determining the machine types and machine routing sequence.
2. Write MATLAB programming codes to reflect minimization of material travelling distances or flow distance score.
3. Apply Genetic Algorithm to continuously iterate and optimize the design of fractal layout and flow assignment according to the performance of the parameters.
4. Quality of the resulting layout is assessed and compared against Fractal cell layout according to (Venkatadri *et al.*1997)

4. THE PROPOSED FRACTAL MANUFACTURING LAYOUT DESIGN

As a guide, the initial fractal manufacturing layout adopted for this study is the cellular manufacturing systems configuration proposed by (Co and Araar 1988) (Figure 5). This layout is re-designed and reconfigured from its initial cellular manufacturing layout using the GA optimization technique. Limitations of the cellular manufacturing layout include inflexibility due to a fixed set of part families, limited allowance for inter-cell flows, and long product life cycles which makes it incapable of performing in unstable environments. It also contains different types of machines which increases the product inter-cell and intra-cell travelling distances.

The design by (Co and Araar 1988) is modified and illustrates the process of constructing a fractal job shop. The example presented is a job shop with 15 distinct product types and 10 types of machine in the initial cellular layout. A total of 64 workstations are proposed by (Co and Araar 1988) in the 6 cells modified group layout design within a factory. But, each group cell contains uncertain numbers of machines. Montreuil *et al.* (1999) propounds that the grouping procedure implements a multi objective mathematical programming formulation with few surrogates;

- Minimize the difference between the assigned workload and capacity available.
- Maximize the number of products that are completed in each cell.
- Maximize the number of cells.

But, it is found that the objectives above are conflicting. The design for the group layout makes the job shop appear very much like a flow shop. But the group layout design suffers from the major disadvantage that requires too many workstation replicates (Montreuil *et al.*, 1999).

In this study, the GA approach lets us represent the entire group layout proposed by (Co and Araar 1988) as chromosomes. The modified group layout by (Co and Araar 1988) is shown in (Figure 5). MATLAB programming has made the representation of the machines in each cell easier. For instance, Cell1 can be represented as (1 5 2 6; 7 4 3 8; 9 10 3 5; 2 10 8 6; 1 5 9 10) in MATLAB codes. Cell4 is coded as (3 9 2 8; NaN NaN NaN 5) (where NaN means Not-a-Number in computing). Cell1 and Cell4 are combined using crossover operations. After the crossover, Cell1 is re-generated and it becomes one of the output cells for fractal manufacturing layout.

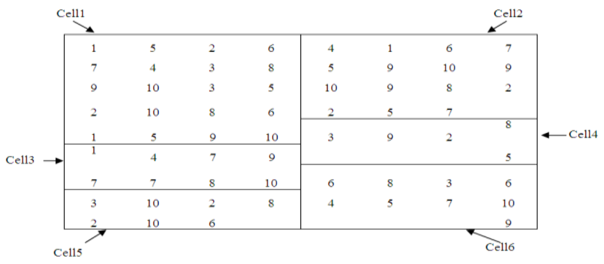


Figure 5: Modified group layout from (Co and Araar 1988)

The fractal manufacturing cell layout proposed by (Co and Araar 1988) has a number of characteristics and is shown in (Figure 6). All fractal cells are similar and contain roughly the same composition of machines. Similarity of fractal cells in terms of machine types and quantities enable high efficiency in controlling shop floor, high operational flexibility and high flexibility for factory expansion. Moreover, all fractal cells are independent and are also capable of processing all products routed to them. Furthermore, products are distributed evenly among fractal cells.

The design of fractal layout (Figure 6) contains three cells. This choice leads to a cell population of 10 workstations, which is within tractable standards of 5 to 15 machines in each fractal cell. It is not necessarily to limit the number of workstations to 30 machines in this case (Venkatadri *et al.*, 1997). But, by adding few more workstations congestion could be alleviated and flow efficiency could further be improved. Therefore, it is logical and reasonable to increase number of Machine 7 in the following approach in the fractal manufacturing layout that is proposed by Venkatadri *et al.* (1997) and Montreuil *et al.* (1999).

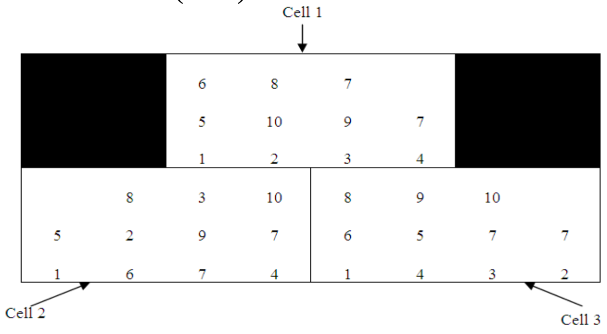


Figure 6: Fractal Manufacturing Layout

4.1. Design Parameters

It is estimated that 10 types of machines are required in the fractal job shop. Machine requirement planning represents the beginning of the fractal layout process. This is carried out by computing the total number of hours required for processing the product demand (Montreuil *et al.*, 1999). There are 15 types of products that are required to be processed in the 3 fractal cells. Based on bottleneck analysis, the total demand for the fractal layout is estimated to produce 400 products that can be processed in the fractal system without violating aggregate capacity constraints and respecting product demands. The other design parameters that are used for the fractal layout modeling have to be defined and calculated as below:

Machine types in fractal job shop = 10
Product types in fractal job shop = 15
Total number of products demand = 400 products
Demand for fractal job shop = 400/15 = 26.67
Total machine processing times = 1108 minutes = 18.47hours
Machine processing times for processing the demand
= 18.47hours x 26.67
= 492hours

Total machine capacity (available hours) is 1297hours
Minimum number of machine required for fractal cell
= Machine capacity ÷ Machine processing times
= 1297 hours ÷ 492 hours
= 2.6 machines = 3 machines

Fractal decomposition is carried out using the procedures outlined in the section on cell creation design. The results of the calculation are shown on (Table 1). It can be shown that 3 machines are required for the 3 fractal cells. Therefore, it is feasible for each type of machine to be replicated or regenerated 3 times. The expected fractal layout contains 30 machines where each fractal cell has 10 machines.

Table 1: Number of replicates for fractal cell layout

Machine Type	1	2	3	4	5	6	7	8	9	10	Total
Number of Replicates	3	3	3	3	3	3	3	3	3	3	30

4.2. Input Data

Tables (2, 3 & 4) contain the input data for the machine processing sequence, process times of products in minutes and machine capacity in hours respectively for each of the replicates (Co and Araar 1988). These data are written in Microsoft excel file and imported into MATLAB programming for the optimization process.

Table 2: Machine routing sequence for 15 types of product

Product Type	Machine Processing Sequence									
1	1	4	7	3	10	8				
2	3	9	2	8	5	6				
3	2	3	4	5	9	10				
4	1	7	8	10	2	3				
5	5	6	8	1	4	7	9			
6	5	2	6	4	1	7				
7	6	4	5	7	10	9				
8	1	3	5	6	8	10				
9	3	4	2	1	5	9	10			
10	8	10	2	4	6					
11	3	1	9	5	7					
12	1	9	10	2	7	8	3			
13	4	3	10	2	8	6				
14	4	2	8	5	1	6				
15	1	5	2	6	8	3	4	7	9	10

Table 3: Machine processing times for 15 types of product

Product Type	Machine Processing Times (Minutes)									
1	10	7	20	15	8	17				
2	10	15	15	15	10	5				
3	11	13	20	15	12	10				
4	9	17	9	8	10	20				
5	9	7	7	15	15	12	9			
6	7	6	13	10	8	8				
7	7	13	12	19	14	13				
8	12	11	18	11	13	10				
9	6	9	8	17	20	12	13			
10	12	18	7	5	6					
11	13	12	9	8	11					
12	7	13	17	6	11	12	5			
13	13	20	5	15	12	17				
14	7	12	20	9	18	8				
15	20	12	13	13	13	5	7	20	7	5

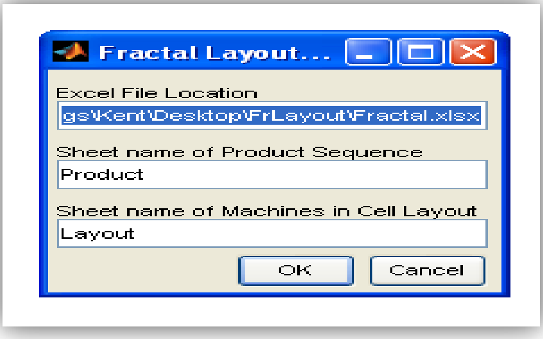
Table 4: Machine capacity for each replicate

Machine Type	Machine Capacity (Hours) for each replicate									
1	25	15	10	30						
2	16	29	15	25	30	20	28			
3	17	15	40	30	10					
4	18	19	17	28						
5	15	20	30	20	20	20	30			
6	18	20	15	15	10	15				
7	10	20	20	10	15	20	15	15	15	10
8	20	20	15	15	10	10	10			
9	18	17	20	30	40	30	20	17		
10	20	10	10	10	30	30	30	15	15	

4.3. MATLAB dialog box

A dialog box is created as an interaction tool on MATLAB. The dialog box pops up to request for input data as shown on (figures 7, 8 & 9). These data are used to verify; the location of Microsoft Excel input file, sheet name of product sequence that is required for the modeling operation, and the sheet name of machines in fractal cell layout; the desired number of fractal cells that are needed; number of rows and columns for each pair of initial cells that are required to generate each fractal cell; he cells required for crossover operation and the desired number of iterations needed for generating the final fractal manufacturing layout.

The input dialog box (Figure 7) for file location and sheet name in Microsoft Excel has been used to ensure the location of the input data is identified and verified. The input dialog box (Figure 8) for desired number of cells is used to insert the number of cells that are required for the initial cell layout. The input dialog box (Figure 9) is for the number of iteration needed to determine number of replicates and analyze the output of the flow distance score.


Fractal Layout...

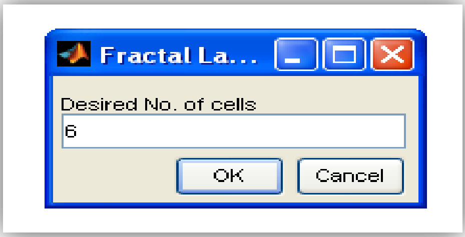
Excel File Location
gs\Kent\Desktop\FrLayout\Fractal.xlsx

Sheet name of Product Sequence
Product

Sheet name of Machines in Cell Layout
Layout

OK Cancel

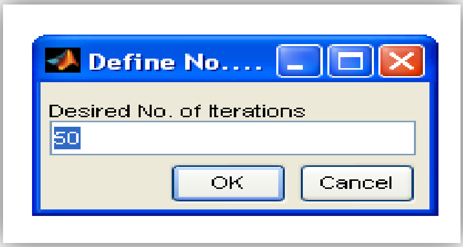
Figure 7: Input dialog box for file location and sheet name in Microsoft Excel


Fractal La...

Desired No. of cells
6

OK Cancel

Figure 8: Input dialog box for desired number of cells


Define No....

Desired No. of Iterations
50

OK Cancel

Figure 9: Input dialog box for number of iteration

4.4. Facility layout problem (FLP)

The FLP is defined as "the determination of the relative locations for, and the allocation of the available space among a number of workstations" (Azadivar and Wang, 2000). The resources could be different sizes and the interactions between resources may vary. This is a major concern in developing a block layout that represents an optimal shape and arrangement of departments within a facility (Hicks 2006). The FLP is a combinational problem for which the optimal solution can be found for small problems. GA based search is one of the good method for dealing with problems of facility layout. In the GA approach to optimization, feasible solution to the problem is encoded in data structures in the form of a string of decision choices that resemble chromosomes. GA maintains population of chromosomes or individuals that are created. The layout design is characterized by chromosomes' fitness which is measured by its value of objective function. Off-springs are created through reproduction, crossover, and mutation (Balamurugan *et al.*, 2006). FLP is formulated as a quadratic set which covers linear integer programming problem, mixed-integer programming problem and graph-theoretical problem. Therefore, quadratic assignment problem (QAP) formulation has been popular in this kind of problems. But manufacturing practice normally requires particular layout configurations such as single row, multi-row or loop formations. These practical constraints place a huge restriction on the optimization process (Hicks 2006), but the GA based search is one good method of dealing with problems of facility layout.

The pick-up and delivery points position of each cell in our study are located on either one of the cell axes (Rajasekharan *et al.* 1998). In this model, the fractal cells are considered to be rectangular blocks with known dimension of (w, h) where w is width and h is height of each cell. After the crossover and mutation, the facility layout for FrMS for this model has a height, h of 3 rows and width, w of 4 columns. If the fractal cells are written as three rows and four columns in matrix form in MATLAB, then the Pick-up Point is (1, 1) and Delivery Point is (3, 4)(Figure 10).

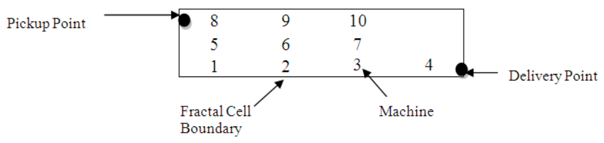


Figure 10: Facility layout problem for FrMS

Some logical assumptions are made for the facility layout problem. These include that the dimensions of the floor area on which the fractal cells are placed is given. The floor space for the flow path on the floor area is not considered. It is also assumed that the flow paths consist of segments that are horizontal and vertical to the walls of the floor (Hu *et al.*, 2007).

The fractal layout dimension, (3 x 4) is chosen because we are considering 10 machines in this study. Thus, we require at least 10 locations for the rectangular fractal cell layout. So, it is feasible to generate a facility layout with 10 machines and 2 spaces. This layout could reduce the material travelling distance by having multi-purpose machines in each fractal cell.

5. IMPLEMENTING THE PROPOSED GENETIC ALGORITHM APPROACH

An iterative algorithm is implemented to optimize the layout and flow assignment according to the design parameters. The layout of each cell is refined using the implied flows between stations. The replicates are re-applied until the heuristic procedures could not find a better solution. The cells are continually iterated to obtain the optimal flow assignment and hence achieve the optimum fractal layout (Montreuil *et al.*, 1999). The GA procedures - selection, crossover, row inverting mutation, column inverting mutation, and deleting mutation are embedded in the iterative procedure in order to generate the optimal material travelling distances. Hence the desired workstation layout that minimizes the material travelling distances and capacity requirements for product demand and mix is created. Each optimal fractal cell is selected based on the flow distance score. Thus, optimum fractal manufacturing layout is created by combining the three optimal fractal cells. The illustrations of the GA steps are presented by showing the first iteration of the fractal cell 1. Initial cellular layout is assumed to contain 6 cells. Fractal cell1 is generated by combining cell 1 and cell 4 by crossover operation. Cell 1 is shown as parent1 and cell 4 is illustrated as parent2 in MATLAB program codes. Chromosomes for each Parent are represented by the various kinds of genes. The genes are represented by the number 1 to 10 that signify that Machine1 to Machine10 are used. Parent1 is represented as (1 5 2 6; 7 4 3 8; 9 10 3 5; 2 10 8 6; 1 5 9 10), illustrated in 5 rows and 4 columns. Parent2, contains 2 rows and 4 columns as (3 9 2 8; NaN NaN NaN 5). The chromosome for each parent is represented in rows. This means that the chromosomes for Parent1 are (1 5 2 6), (7 4 3 8), (9 10 3 5) and so on. One of the chromosomes from Parent1 is chosen randomly. For instance, the first row chromosome for Parent1 has been selected for the crossover function. On the other hand, the 1st row chromosome for Parent2 also is selected to be combined with the chromosome of Parent1 as shown in (Figure 11). The continuous selection of the chromosomes for Parent1 and Parent2 generated 10 different Offspring after the crossover operation (Figure 11). Two Off-springs are generated from each iteration of the crossover. The Offspring1 that is created from selection and crossover with 5 chromosomes are selected for the upcoming mutation. Offspring2 is not been used because there are only 3 chromosome lesser than Offspring1.

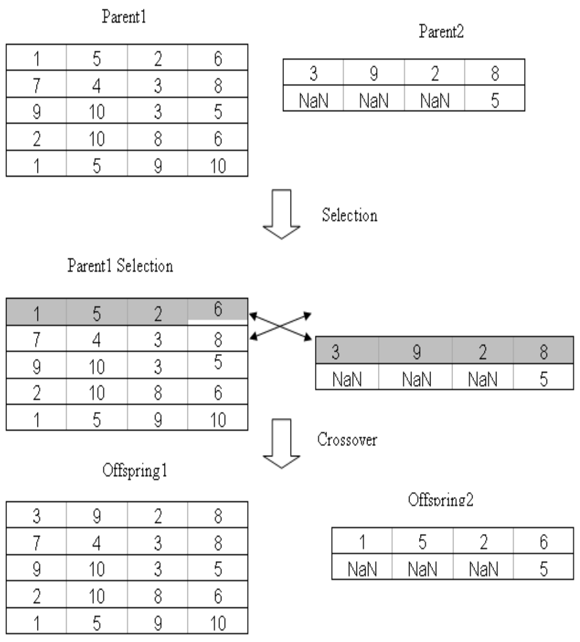


Figure 11: Selection and Crossover

Inverting mutation takes place after the crossover. The Offspring that is generated in the previous crossover is used as the Parent again in this inverting mutation operation. Initially, a cutting point is randomly introduced anywhere along the last row of the Parent. The cutting point indicates the row of the chromosomes for the inverting mutation. The last row of the chromosome is being mutated to the initial row based on the programming code "circshift" - (mathscript function). The iterations of the row inverting mutation are replicated four times as shown in (Figure 12). For each offspring that is generated, three column inverting mutations take place. For column inverting mutation, chromosome is represented column by column. The cutting point is set in the last column of the chromosome. The column based chromosome is mutated and shifted from the last column to the first column. After this, the Parent is replicated by shifting its chromosomes in columns as shown in (Figure 13). For each Parent that is obtained from the previous mutation step, the entire inverting mutation is expected to replicate 12 times.

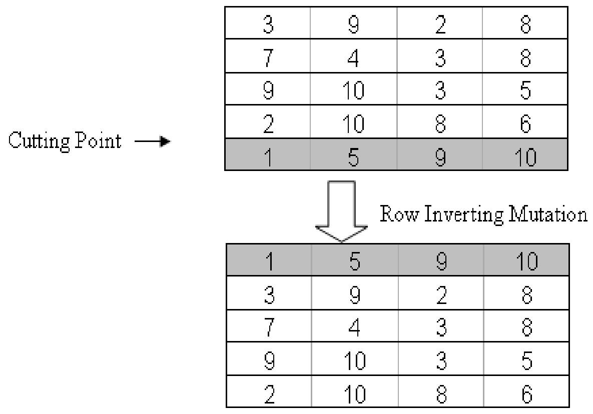
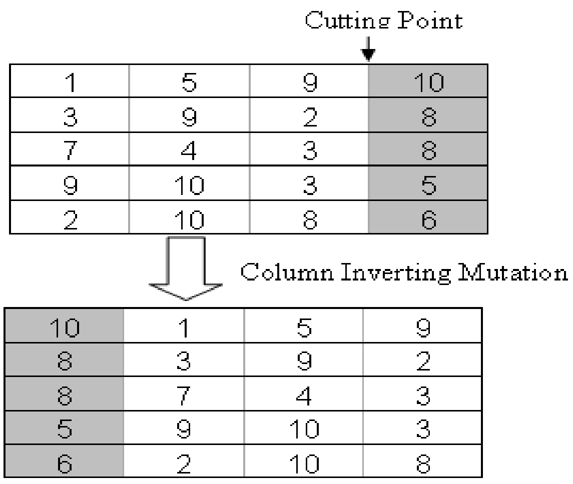


Figure 12: Row Inverting Mutation

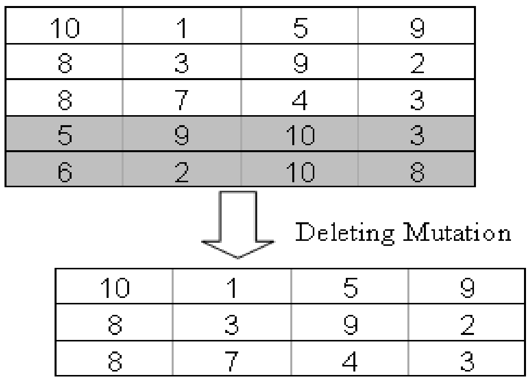


1	5	9	10
3	9	2	8
7	4	3	8
9	10	3	5
2	10	8	6

10	1	5	9
8	3	9	2
8	7	4	3
5	9	10	3
6	2	10	8

Figure 13: Column Inverting Mutation

After inverting mutation, the Child is generated and transformed to be the Parent again for deleting mutation as shown in (Figure 14). On completion of the previous mutation, the process of deleting mutation is simplified by just deleting the last two rows of the five chromosomes in the Child.



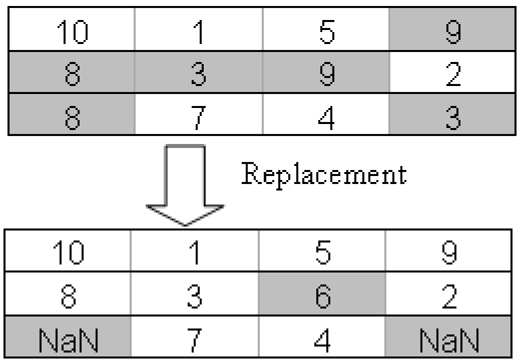
10	1	5	9
8	3	9	2
8	7	4	3
5	9	10	3
6	2	10	8

10	1	5	9
8	3	9	2
8	7	4	3

Figure 14: Deleting Mutation

Replacement is the last step in the process of generating fractal cell layout as shown in (Figure 15). In fact, each fractal cell requires 10 machines where no duplicated machines or missing machines are allowed. This is because duplicated machines will increase the material travelling distance. Minimum flow distance score is the requirement for fractal cells.

As a result, machine3, machine8 and machine9 are grouped as duplicated machines that required to be replaced by missing machines. The MATLAB codes are programmed to search the missing machines. The missing machine in this scenario is machine6. Thus, machine6 replaces one of the duplicated machines.



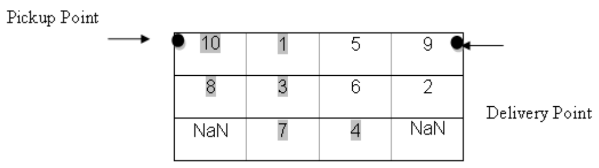
10	1	5	9
8	3	9	2
8	7	4	3

10	1	5	9
8	3	6	2
NaN	7	4	NaN

Figure 15: Replacement

The fractal cell layout that is generated after Replacement can be represented as (10 1 5 9; 8 3 6 2; NaN 7 4 NaN). From the Facility Layout Problem (FLP) that was discussed in the previous section, materials are moved into the cell through Pick-up Points and moved out from the cell through Delivery Points as shown in (Figure 16). The Pick-up Point is at (1, 1) while the delivery Point is at (3, 4).

The fractal cells are capable of processing all 15 types of product. Therefore, the materials to be produced need to be processed in specified machine routing sequence. For instance, materials that are used to produce Product1 need to be processed by machine1, machine4, machine7, machine3, machine10, and machine8 in continuous sequence. Each location of machines is represented on (x, y) coordinates. Before the materials are processed in machine1, they have to be carried into the fractal cell through the Pick-up Point. After processing in all the machines within the fractal cells, the final product1 gets delivered to the shipping department through Delivery Point as shown in (Figure 16).



10	1	5	9
8	3	6	2
NaN	7	4	NaN

Figure 16: Material Routing sequence for Product1

Then the flow distance score is calculated based on the mathematical solution in MATLAB which is represented as:

Distance = abs (buffer1 (1)-buffer2 (1)) + abs (buffer1 (2)-buffer2 (2)) (1)

The abs is representation of absolute. The absolute value allows the distance to the left (negative value) and distance to the right (positive value) to be counted into the total distance. Buffer1 and buffer2 is the matrices of data that are being stored in temporary memory.

The shortest routing distance is always considered from the various iterations that are being generated for each of the fractal cell.

6. OUTPUT RESULTS AND DISCUSSIONS

The computational result of product travelling distances within the fractal cells indicates the flow scores of fractal layout. Flow score is computed and represented as the product travelling distances.
The optimal fractal layout with the minimum flow distance scores is selected by MATLAB and displayed. These output data are used to draw the graphs of flow scores with different generations and flow scores with different product ranges. The GA search for an optimal solution yielded results from 100 iterations and the output is converted into the final fractal cell layout representing the fractal manufacturing layout. The material travelling distances for each of the three fractal cells work out as follows in terms of flow distance scores;

Flow distance score for Cell 1 = 205
Flow distance score for Cell 2 = 217
Flow distance score for Cell 3 = 197

Overall flow distance score for the final fractal manufacturing layout through the proposed GA = 619 and this is shown on (Figure 17).

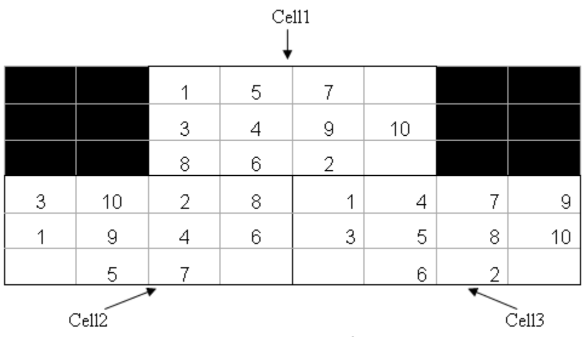


Figure 17: Final Fractal Manufacturing Layout A

Comparatively, the fractal layout according to (Venkatadri *et al*.1997) has machine requirements similar to our final layout requirements with the following flow distances;

Flow distance score for Cell 1 = 251
Flow distance score for Cell 2 = 252
Flow distance score for Cell 3 = 257

Overall flow distance score for Final Fractal Layout according to (Venkatadri *et al*.1997) is = 760 and that is shown on (Figure 18).

This shows that the flow distance score obtained from the proposed GA approach is lesser at 619 than that of (Venkatadri *et al*.1997).

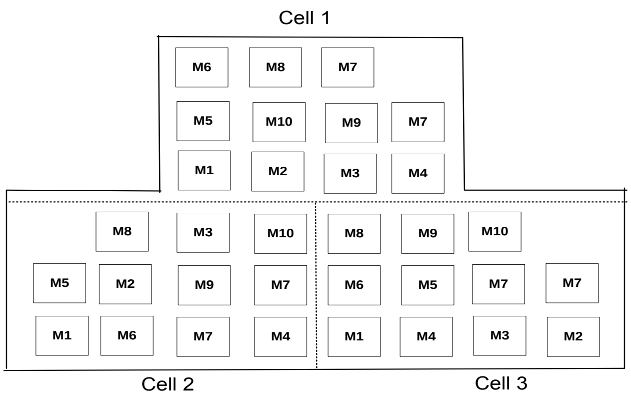


Figure 18: Fractal cell layout according to (Venkatadri et al.1997)

Ascertaining or working out the optimal number of iterations in each cell for our proposed GA approach aided in producing the right flow distances and involved plotting flow distance score against iterations as shown on figures (19), (20) and (21) for cells 1, 2 & 3. These plots signify the optimal flow distances at 205, 217, and 197 for cells 1, 2, & 3 respectively.

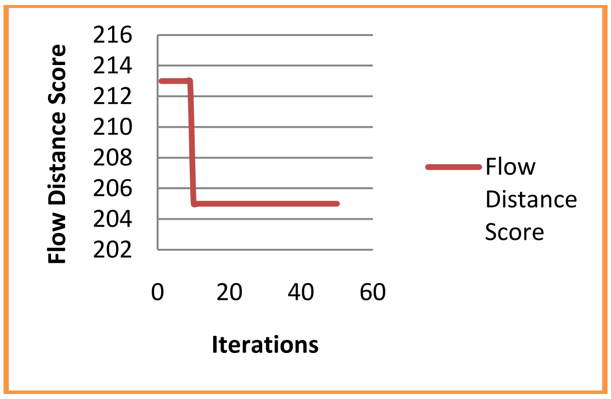


Figure 19: Flow distance score for fractal cell 1

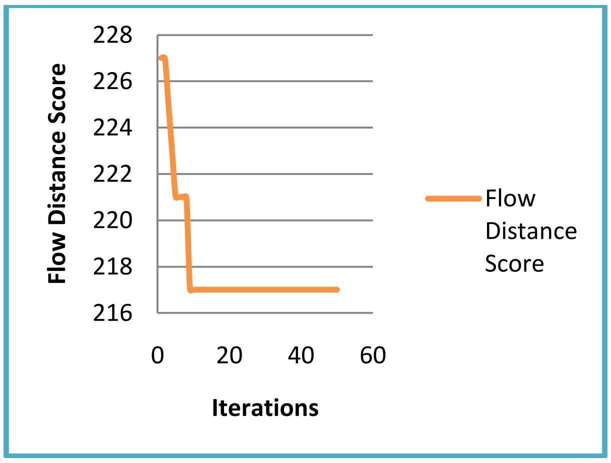


Figure 20: Flow distance score for fractal cell2

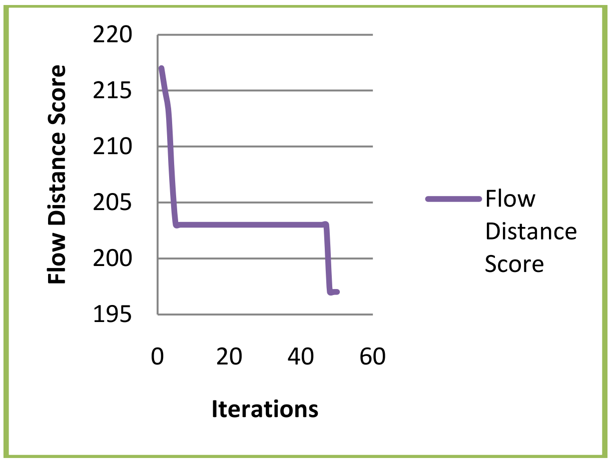


Figure 21: Flow distance score for fractal cell3

7. CONCLUSION

The GA approach has been applied in the design of the fractal manufacturing shop floor layout. This algorithm was used to search for the optimal fractal cell layout for efficient and effective material/ product movements within the shop floor. Fundamentally, the decision of how to distribute/assign products to cells as evenly as possible to aid responsiveness to uncertainties in manufacturing and easy control of resources was seen to be very important to the design, implementation and final outcome of the experimentation. The model implemented using MATLAB managed the scenario quite well and handled the mathematical formulations, swapping and deleting matrices etc. quite efficiently. Overall, the computational results indicated that unrestricted product flows offer the best flow scores in a fractal layout.

REFERENCES

Al-Sultan, K, S., and Fedjki, C. A., 1997, A Genetic Algorithm for the Part Family Formation Problem, *Production Planning & Control*, Vol. 8, No.8, pp. 788-796.

Askin, R. G., Ciarallo, F. W. and Lundgren, N. H., 1999, An empirical evaluation of holonic and fractal layouts. *International Journal of Production Research*, 37, 5, 961-978.

Askin, R. G., Lundgren, N. H. and Ciarallo, F., 1996, A material flow based evaluation of layout alternatives for agile manufacturing. In R. J. Graves, L. F. McGinnis, D. J. Medeiros, R. E. Ward and M. R. Wilhelm (eds.), *Progress in material handling research* (Braun-Brumfield), pp. 71-90.

Azadivar, F., and Wang, J., 2000, Facility Layout Optimization using Simulation and Genetic Algorithms, *International Journal of Production Research*, Vol. 38, No. 17, pp. 4369-4383.

Balamurugan, K., Selladurai, V., and Ilamathi, B., 2006, Design and Optimization of Manufacturing Facilities Layouts, *Institution of Mechanical Engineers*, Vol. 220, Part B, pp. 1249-1257.

Chase, R. B. and Aquilano, N. J., 1992, *Production & operations management - A life cycle approach*, 6th Ed. (Irwin: Boston, Massachusetts, USA).

Co, H. C. and Araar, A., 1988, Configuring cellular manufacturing systems. *International Journal of Production Research*, 26, 9, 1511-1522.

Gau, K. Y. and Meller, R. D., 1999, An iterative facility layout algorithm. *International Journal of Production Research*, 37, 16, 3739-3758.

Hicks, C., 2006, A Genetic Algorithm Tool for Optimization, Cellular or Functional Layouts in the Capital Goods Industry, *International Journal of Production Economics*, Vol. 104, pp. 598-614.

Holland, J. H., 1975, Adaptation in Natural and Artificial Systems, University of Michigan Press, Michigan.

Hu, G. H., Chen, Y. P., Zhou, Z. D., and Fang, H. C., 2007, A Genetic Algorithm for the Inter-Cell Layout and Material Handling System Design, *The International Journal of Advanced Manufacturing Technology*, *Springer*, Vol. 34, pp.1153-1163.

Kamrani, A. K., and Gonzalez, R., 2003, A Genetic Algorithm-Based Solution Methodology for Modular Design, *Journal of Intelligent Manufacturing*, Vol. 14, pp. 599-616.

Kiusalaas, J., 2005, *Numerical methods in Engineering with Python*, Cambridge University press.

Montreuil, B., Venkatadri, U. and Rardin, R. L., 1999, Fractal layout organization for job shop environments. *International Journal of Production Research*, 37, 3, 501- 521.

Parames Chutima, 2001, Genetic Algorithm for Facility Layout Design with Unequal Departmental Areas and Different Geometric Shape Constraints, *Thammasat International Journal Science and Technology*, Vol. 6, No. 2, pp. 33-43

Rajasekharan, M., Peters, B. A. and Yang, T., 1998, A Genetic Algorithm for Facility Layout Design in Flexible Manufacturing Systems, *International Journal of Production Research, Taylor & Francis Ltd.*, Vol. 36, No. 1, pp. 95-110.

Saad, S. M. and Lassila, A.M., 2004, Layout design in fractal organizations, *International Journal of Production Research*, Vol. 42, Vol. 17, p. 3529-3550.

Venkatadri, U., Rardin, R. L. and Montreuil, B., 1997, A design methodology for fractal layout organization. *IIE Transactions*, 29, 911-924.

Xiaodan W., Chao-Hsien C., Yunfeng W., and Weili Y., 2007, A Genetic Algorithm for Cellular Manufacturing Design and Layout, *European Journal of Operational Research.*, Vol. 181, No. 181, pp. 156-167.

Wild, R., 1993, *Production and operations management*. 4th Ed. (Cassell: London)