OPERATIONAL DESIGN OF A CELLULAR MANUFACTURING SYSTEM

By

Adnan Tariq

Submitted for the Partial Fulfillment of the Degree Of
Doctor of Philosophy

Supervisor
Prof Dr Abdul Ghafoor

Co-supervisor
Prof Dr Iftikhar Hussain

Mechanical Engineering Department
College of Electrical & Mechanical Engineering (E & ME)
National University of Science & Technology (NUST)
Rawlpindi, Pakistan

2010

Form PhD-5
DOCTORAL PROGRAMME
OF STUDY
(Must be type written)

National University of Sciences & Technology, Rawalpindi

DOCTORAL DEFENCE

We hereby recommend that the student Mr Adnan Tariq

Regn No 2002-NUST-TfrPhD-Mech-21 may be accepted for Doctor of Philosophy Degree.

DOCTORAL DEFENCE COMMITTEE

Held on 07 April 2010

GEC Member 1:	Brig Dr Nawar Khan	Signature :	[illegible]
GEC Member 2:	Prof Dr Muhammad Afzaal Malik	Signature :	M. Afzaal Malik
GEC Member 3 :	Col Dr Javaid Iqbal	Signature :	[illegible]
Supervisor:	Brig Dr Abdul Ghafoor	Signature :	[illegible]
Co-Supervisor: (if appointed)	Prof Dr Iftikhar Hussain	Signature :	[illegible]
External Evaluator 1:	Prof Dr Sahar Noor, U.E.T Peshawar	Signature :	Sahar

Foreign Evaluators

Evaluator 1:	Dr Khurshid Khan, UK	Signature :	Letter attached as Anx A
Evaluator 2:	Dr Sheikh Meeran, UK	Signature :	Letter attached as Anx B

COUNTERSIGNED

Dated: 07 Apr 2010

[illegible]
Dean/Commandant/Principal/DG

Dr. Akhtar Nawaz Malik
Dean/Professor
College of E&ME (NUST)
Rawalpindi PAKISTAN

Distribution:

- 1 x copy each to Registrar, D(R&D), D(E&A) at HQ NUST and HoD, Supervisor, Co-Supervisor (if appointed), in student's dossier and each member of GEC.

OPERATIONAL DESIGN OF A CELLULAR MANUFACTURING SYSTEM

Keywords: Group Technology; Cellular Manufacturing; Machine-Part grouping; Genetic Algorithms; Grouping Efficacy; Local Search Heuristic; Job-Shop; Scheduling.

ABSTRACT

Cellular Manufacturing (CM), which contains the flexibility of Job-Shop and at the same time has a higher rate of production as flow lines, is proving to be a useful substitute for the production carried out in batches. In spite of the fact that there are so many benefits associated with CM but designing CM, for real world problems, is a very complex job. Since the main task in designing a CM is grouping of machines into cells and parts into corresponding families, therefore, most of the research carried out so far has considered the Cellular Manufacturing System (CMS) design as a Machine-Part grouping problem only and focus on the operational aspects of the design has been very little. Once the Machine-Part grouping stage is over, scheduling of the system is supposed to be the next stage in completing the operational design of a CMS. This is the stage where important production related information; such as processing sequence and processing time is taken into consideration. Scheduling is very essential as it enhances productivity and maximizes the usefulness of a given manufacturing system by utilizing the available resources in an optimized manner. Therefore, alongside Machine-Part grouping, scheduling is of paramount importance too, as it ensures proper utilization of resources.

In order to carryout a complete operational design of CMS, a two stage methodology has been developed in this research. First, the problem of Machine-Part grouping (CMS design) is solved, and then sequencing and scheduling of parts on machines is carried out. Since each cell is like a Job-Shop, therefore the scheduling part of the problem is solved using a similar approach as in case of a Job-Shop scheduling problem (JSSP).

Separate hybrid tools, for solving Machine-Part grouping problem and Job-Shop Scheduling Problem (JSSP), has been developed by combining Genetic Algorithms (GA) with Local Search Heuristics (LSH). Each tool's effectiveness has been verified, separately, by solving a number of benchmark problems from literature. Finally, the two tools are combined in such a manner that the output of the Machine-Part grouping serves as an input to the tool developed for the scheduling of Job-Shop. Final outcome of the program is a cellular arrangement of the system (machine groups and corresponding part families) and detailed information about the sequencing and scheduling of the system.

The development of two effective hybrid GA based tools, for Machine-Part grouping and Job-Shop Scheduling, and their combination are the main contributions of this research.

ACKNOWLEDGEMENTS

I am thankful to Almighty Allah who gave me the strength and wisdom to complete this research work.

It has been a long journey to finally complete one of my dreams. Many people have been involved in this journey. I must admit that without their support and guidance, it would have been much more difficult to accomplish this success. At this time, I would like to thank each of them for their encouragement and support.

I would like to express my thanks and appreciation to my supervisor, Brig Dr Abdul Ghafoor, for his continuous support and advice that helped me complete this dissertation.

I would also like to express my special thanks and appreciation to my co-supervisor Prof Dr Iftikhar Hussain (Chairman Industrial Engineering Department, NWFP University of Engineering & Technology, Peshawar, Pakistan) who extended a great deal of help and guidance in the completion of this research work.

I would also like to acknowledge the consistent support and help of my Guidance & Evaluation Committee (GEC) members; Prof Dr Nawar Khan, Dr Javed Iqbal, and Prof Dr Afzaal Malik.

I am extremely thankful to my foreign experts (Dr M. Khursheed Khan, Bradford University, UK, and Dr Sheikh Meeran, Bath University, UK) and local expert (Dr Sahar Noor, NWFP UET, Peshawar) for their valuable comments and suggestions that helped me improve this thesis a great deal.

I am also thankful to Higher Education Commission (HEC), Pakistan, for their financial support during this research work.

Finally I would like to thank all my family members and friends for their love and support during my studies.

In the end I would like to dedicate this research work to my late father, who would have been feeling extremely proud while seeing me achieving this goal.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

GE_T	Accumulative GE of a whole population
ART	Adaptive Resonance Theory
ASC	Additive Similarity Coefficient
Cum Prob[*x*]	Array for cumulative probability of each chromosome in population
GE [*x*]	Array for grouping efficacy of each chromosome in population
NM [*k*]	Array for number of machines in cell *k*
NP [*k*]	Array for number of parts in cell *k*
Sel Prob [*x*]	Array for selection probability of each chromosome in population
AI	Artificial Intelligence
ANN	Artificial Neural Network
Best [*x,y*]	Best chromosome of the population
Best [*x*]	Best chromosome of the population
BEA	Bond Energy Analysis
BB	Branch and Bound
CM	Cellular Manufacturing
CMS	Cellular Manufacturing System
Chrom	Chromosome
CI	Cluster Identification
CT	Completion Time
CPM	Critical Path Method
CR	Critical Ratio
X_1, X_2	Crossover points
DCA	Direct Clustering Analysis
EDD	Earliest Due Date
EST	Earliest Start Time
a_{ij}	Entry at i^{th} row and j^{th} column (either 1 or 0)
ES	Expert System
FIFO	First In First Out
Fuzzy ART	Fuzzy Adaptive Resonance Theory
FL	Fuzzy Logic
Gen	Generation number
GA	Genetic Algorithm
GERT	Graphical Evaluation and Review Technique
GT	Group Technology
GE	Grouping Efficacy
IC	Intercellular
Chrom [*i,j*]	Is a two dimensional array for population
Excess[*w*]	Is an array for those integers which are in excess
Short [*z*]	Is an array for those integers which are in shortage
E. Amnt [*w*]	Is an array showing the excess amount of each integer which is in excess
S. Amnt [*z*]	Is an array showing the shortage amount of each integer which is in shortage
JAT	Job Availability Time
JSSP	Job-Shop Scheduling Problem
LSH	Local Search Heuristic
LB	Lower Bound
MAT	Machine Availability Time
MP_{ini} [*i,j*]	Machine-Part incidence matrix

C_{max}	Makespan
MP	Management Priority
Max GE	Maximum Grouping Efficacy
Max Gen	Maximum Number of Generations
MST	Minimum Spanning Tree
MSC	Multiplicative Similarity Coefficient
NC	Number of Cells
PT	Processing Time
PERT	Project Evaluation and Review Technique
ROC	Rank Order Clustering
$\text{Child}_A\,[i,j]$	Represents child A that results after the crossover procedure
$Chrom_A[i,j]$	Represents the A^th chromosome in the array for population (Pop $[x, y, z]$)
SOFM	Self Organizing Featured Maps
SPT	Shortest Processing Time
SA	Simulated Annealing
$MP_{fin}\,[i,j]$	The rearranged Machine-Part matrix
Pop $[x, y, z]$	Three dimensional array for population
TBE	Total Bond Energy
N_0^{in}	Total Number of 0s inside the block diagonal
x_k	Total number of 1s in cell *k*
N_1	Total number of 1s in the machine parts incidence matrix
N_1^{in}	Total number of 1s inside the block diagonal
N_1^{out}	Total number of 1s out side the block diagonal
Machs	Total number of machines
Parts/Jobs	Total number of parts/jobs
Pop Size	Variable for population size

CHAPTER 1

INTRODUCTION

1.1 Introduction:

Modern manufacturing systems have been kept under constant pressure by the unpredictability in demand and the ever decreasing product life cycles and are finding it hard to cope with these challenges. That is the reason that Cellular Manufacturing (CM) is seen by many as a promising alternative which provides some immediate benefits of reduction in the costs related to material handling, setup times and work in process. Therefore a Cellular Manufacturing System (CMS) is comparatively well equipped to face the challenges mentioned above.

This Chapter briefly describes Group Technology (GT), the conventional manufacturing systems and finally CMS. The description of different manufacturing systems is followed by the objectives of this research and methodology to be adopted to achieve those objectives. An outline of the report is also given in the end.

1.2 Group Technology (GT):

GT in fact is the philosophy that is based upon doing those things in a similar fashion that are alike (Askin and Standridge [1993]). According to Selim & Askin [1998] "GT is a manufacturing philosophy that groups parts into families by taking into consideration the similarities among parts in terms of design and manufacturing". When the philosophy of GT is implemented into manufacturing, then at the initial stage of design, the components of a product, which are similar to each other, are processed on similar processing arrangements (Irani [1999]). Also, such products are assembled following a similar sequence (Irani [1999]). GT in fact is a source of improving productivity by bringing together the similar recurrent activities and organizing common tasks alongside each other. With the help of GT, a set of parts can be broken down into different part families. Within each family the processing requirements of each part are the same. Manufacturing cells are organized for these part families by grouping dissimilar machines, that are required for processing by each part family, on the production floor. The associated improvements with the implementation of GT are summarized by Pullen [1976], Houtzeel and Brown [1984], and Wemmerlov and Hyer [1989] as follows:

a) Throughput time is reduced from 5% to 90%
b) Reduction in the inventory levels of work-in-process is from 8% to 80%.
c) Savings in terms of materials handling costs are from 10% to 83%
d) The job satisfaction levels are between 15% to 50%
e) The reduction in Fixtures requirements is between 10% to 85%
f) Setup time requirements are reduced between 2% to 95%
g) The reduction in Space requirement stays between 1% to 85%
h) Quality improvements from 5% to 90% can be experienced
i) Finished goods (10-75%)

These benefits result in a substantial decrease in manufacturing cost and increase in product quality. These are the reasons that GT has been so attractive and found successful with medium variety and medium volume production environments.

Before further elaboration of GT and CMS, first the traditional manufacturing approaches (Job-Shops and Flow Lines) are described in the following Sections.

1.3 Job-Shop Manufacturing:

Job-Shop manufacturing systems can be very commonly found in the USA (Black, J. [1991]). The main aim of Job-Shop manufacturing is to achieve a higher degree of flexibility so that products having a wide range of variation in size and shape can be produced, in small lot sizes, in a single facility. The distinguishing feature of Job-Shop is the manufacturing of products that may be having different processing sequences and variation in processing times. Products travel through the entire facility in batches. In Job-Shop environment the main dictating force in selection of machines is variety of products and smaller lot sizes. This is the reason that in Job-Shop manufacturing general purpose machines are mainly utilized as they can perform a variety of operations. The grouping of machines in Job-Shop environment is carried out on the basis of functions e.g. lathe machines are placed in one shop, milling machines in another and so on, as shown in Figure 1.1. That is why a Job-Shop layout is generally termed as a functional layout.

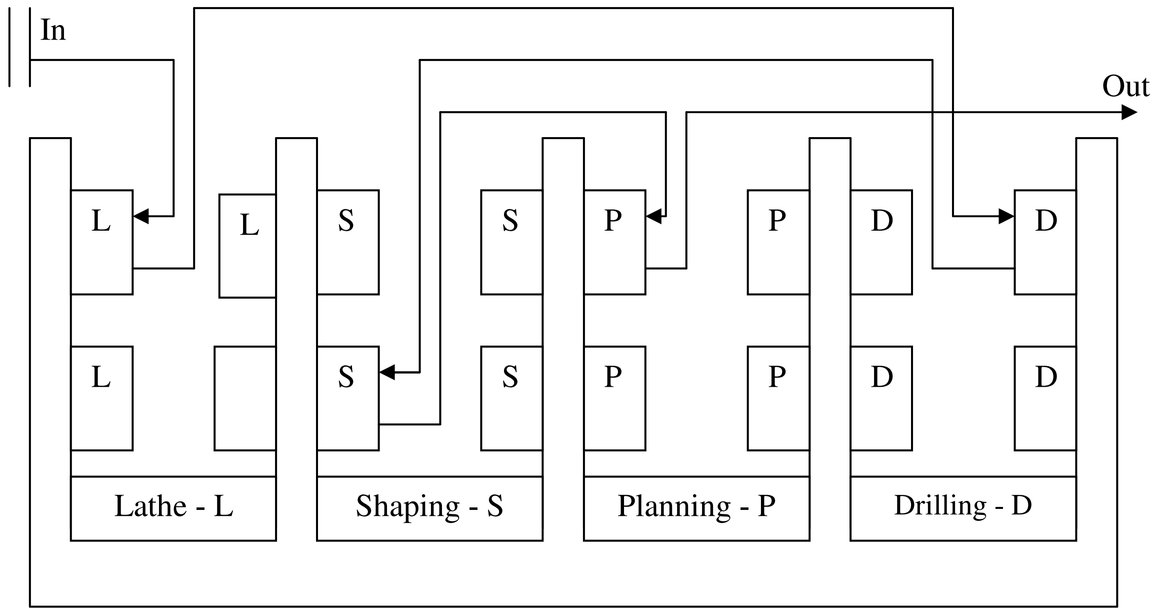


Figure 1.1: Process Based Layout

Mungawattana [2000] presented a comparison of different manufacturing systems. According to him in Job-Shop manufacturing jobs spend less time in processing and most of their time is consumed either waiting in queues or in other non-productive activities for example handling. Since in Job-Shop environment machines are distributed on the basis of their function therefore during processing some jobs have to travel through the entire facility. Therefore, in order to make the processing more economical jobs are processed and moved through the facility in batches which result in longer cycle times and high level of in-process inventory. This results an increase in the cost of production and a decrease in the rate of production.

1.4 Flow Line Manufacturing:

Mungawattana [2000] also observes that the distinguishing feature of Flow Line Manufacturing is its higher rate of production and lower manufacturing cost. Unlike Job-Shop manufacturing here specialized machines are used. The organization of a Flow Line is in accordance with the processing requirements of a product. Once organized, then each flow line is fully dedicated for the manufacturing of a particular product. The presence of specialized machines and their organization according to the processing requirements of a product allow the flow of one piece at a time which results in increase in production rate and decrease in manufacturing time and cost as lesser amount of time is spent by jobs while waiting in queues and handling

[Mungawattana, (2000)]. Lack of flexibility is one major drawback of flow line manufacturing. Main reason for this is the presence of specialized machines which are very expensive and reconfiguration of such machines is normally not allowed.

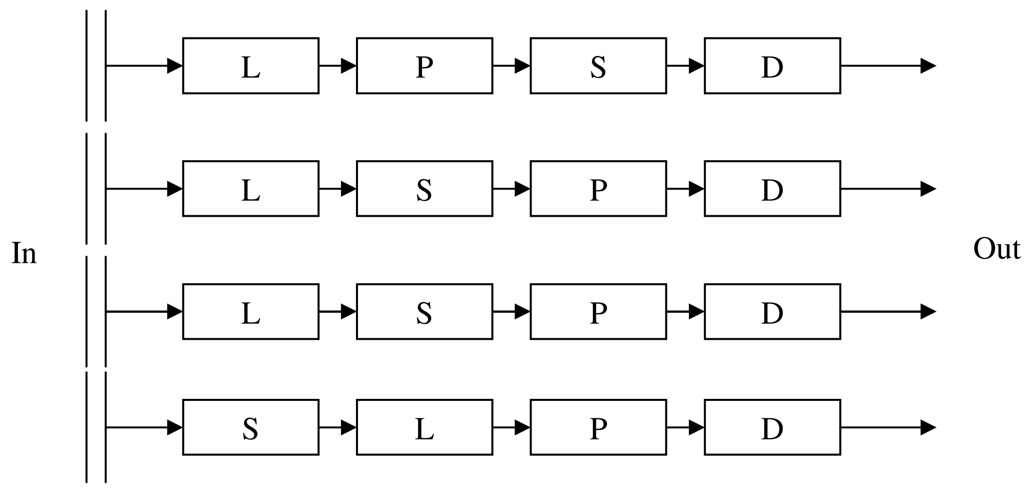


Figure 1.2: Product Based Layout

1.5 Cellular Manufacturing:

From the above discussion it can be concluded that Job-Shops and Flow Lines are comparatively less equipped to meet the present day's challenges that include constant changes in product design, product demand and corresponding reconfigurations in the manufacturing systems. Therefore, CM which is a manufacturing philosophy based on GT, is seen as a promising solution for the problems faced by the present day manufacturing systems. The formation of a CMS mainly consists of two important tasks: grouping of parts into families on the basis of their similar designs and processing requirements and grouping of machines into cells according to the processing requirements of corresponding part families. A group of parts can be called as a family if either their processing requirements are similar or they resemble each other in terms of size and geometric shape (Ham et al. [1985], Groover [2008]). Similarly, a manufacturing cell consists of a group of machines that are dissimilar to each in terms of their functioning and dedicatedly involved in the processing of a part family (Mungawattana, [2000]). Machines in each cell are placed in close proximity to each other (Figure 1.3) thus saving time and cost (handling). Each cell is ideally responsible for the manufacturing of a particular part family which results in simplifying the flow of material and scheduling of the system. In contrast to Job-Shop parts in CM have to travel less distances before their processing is completed. Also, having machines in close proximity the flow of one piece at a time is possible thus saving a lot of waiting time, which is unavoidable in case of Job-Shop manufacturing.

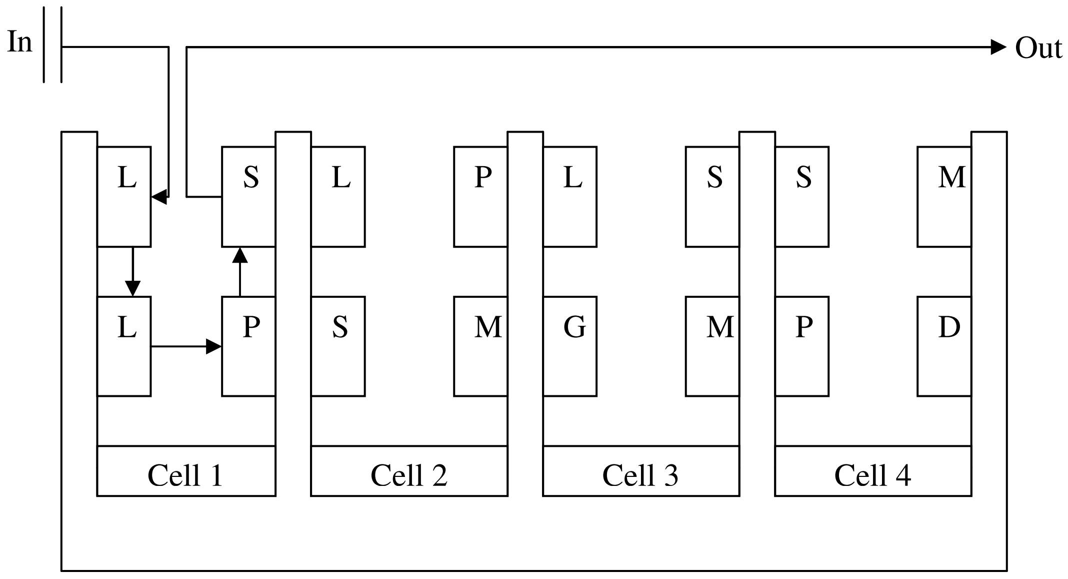


Figure 1.3: Cellular Type Layout

Another aspect of CM that causes a reduction in the over all production time is reduced setup times. It is because of the fact that each part family contains parts that have similar design attributes.

CM in fact provides a system that has the combined advantages of both Job-Shop and Flow Line Manufacturing. Similar to Job-Shop CMS also utilizes general purpose machines and therefore has the ability to be reconfigured and produce a variety of products. Also, having machines in close proximity in each cell and dedicated to a particular part family efficient flow of material and higher rate of production, like a Flow Line Manufacturing system, can be achieved. Finally it can be concluded that wherever there is a requirement of producing a medium variety of products in medium quantity then CM can prove to be, comparatively, more economical, (Black J. [1983]). In case where large volumes are to be produced then pure Flow Line Manufacturing is preferable. Similarly, in case where greater variety of products is to be produced then pure Job-Shop Manufacturing can be more useful. CM over the years has been gaining popularity. Fry et al [1987] observed that several US based manufacturers adopted CM instead of the conventional Job-Shop Manufacturing.

1.6 Advantages of Cellular Manufacturing:

There are many advantages that have been associated with CM. A number of research studies have been carried out in validating and establishing these benefits associated to CM. Some of the well known studies include Greene & Sadowski [1984], Chandrasekharan & Rajagopalan [1987], Askin & Standridge [1993], Shafer & Charnes [1994], Suresh & Meredith [1994], Singh & Rajamani [1996], and Annan Mungawattana [2000]. These advantages are briefly summarized below:

- **Smaller setup times are required:** Since each manufacturing cell is assigned a family of parts that contains parts of similar design attributes, therefore during processing they require similar fixtures and work holding devices. This reduces setup time.
- **Smaller lot sizes can be processed:** Since machines are in close proximity and arranged according to the processing requirements of a particular part family, therefore processing of parts in smaller lot sizes is possible and economical too.
- **Smaller inventory levels both in terms of in-process and finished items:** Due to efficient and smooth flow of material, smaller lot sizes and setup times, the level of inventory both in terms of in-process and finished items is reduced. Another aspect of this efficient material flow is the possibility of producing parts either Just-In-Time (JIT) or in small lot sizes
- **Reduction in time and material handling cost:** Ideally in CM each part family is completely processed inside a particular cell. It saves a lot of time (travel + waiting) and material handling cost which is a regular feature of the Job-Shop environment.
- **Decrease in flow time is observed:** The efficient and smooth flow of material through the system reduces the time both in terms of setup and handling of material which as a whole decreases the overall flow time.
- **Reduction in the number and types of tools required:** In CM each part family consists of parts which have similar shapes, sizes and processing requirements. This is the reason that similar tools are required by almost all the jobs during processing.

- **Space requirements are reduced:** Because of reduction in lot sizes and setup times there occurs a reduction in the inventory levels of both in-process and finished items, which consequently reduces the space requirements.
- **Decrease in throughput times:** In Job-Shop environment parts get processed in almost the entire facility and thus have to travel through long distances. On the contrary in CM each part family is processed ideally inside a particular cell which means parts have to travel through short distances and spend less time in waiting. This reduces the overall throughput time.
- **Improvement in the quality of products:** In CM parts get processed in small lot sizes and on machines which are placed in close proximity. Therefore, during processing, any mistake can be quickly identified and corrected. This leads to improvement in the quality of products and reduction in wastage of material.
- **Improvement in operations' control:** Since in CM jobs are processed inside their respective cells therefore the overall control of operations (scheduling and material) is much easier as compared to Job-Shop manufacturing where mostly jobs have to travel the entire facility while getting processed.

1.7 Problem Definition:

In spite of the fact that CM is very beneficial, a number of CMS design approches - such as Classification and Coding (Hyer & Wemmerlov [1989], Offodile [1991]), Array Based Clustering (King [1980], Chandrasekharan & Rajagopalan [1986], Chu & Tsai [1990]), Graph Partitioning (Kumar et al. [1986], Vohra et al. [1990], Ng [1992 & 1993]), Similarity Coefficient Approach (Lozano et al. [1999], Yasuda & Yin [2001], Yin and Yasuda [2006]), Mathematical Programming (Heragu [1999], Zhao & Wu [2000], Nsakanda et al [2005]), Artificial Intelligence (Peker & Fernando & Mauricio [2002], Kara [2004], Li et al. [2007], Safaei et al. [2008]), Heuristic Based Approaches (Geoffrey et al. [1992], Caux et al. [2000]) - proposed over the years still have major shortcomings. Almost all the techniques so far developed take the CMS design problem as the part machine clustering problem only and very little consideration has been given to important production related information such as production volume, processing sequence, processing times etc. That is why the portability of such techniques into practice is limited and this is causing an increase in gap between practice and research. The linking of CMS design problem with the cell scheduling makes it more useful in practice. But this aspect of linking CMS design with cell scheduling has been very rarely handled in research. Some researchers (Onwubolu [2000], Venkataramanaiah [2006], Celano et al. [2007], Lin et al. [2008]) did focus on the scheduling aspect of the CMS. But the drawback of these approaches is that their entire focus is on cell scheduling only and the CMS design problem has not been considered at all. This shows that the two problems (CMS design and cell scheduling) have been frequently dealt with separately rather than in a sequential/integrated manner. Therefore, the motivation of this research is to develop a methodology that would not only solve the CMS design problem as Machine-Part grouping problem, but also schedule the system by considering some of the important production related information i.e. processing sequence of each part and processing time for each operation of each part on each machine. The system would be initially provided with an input in the form of a 1-0 Machine-Part incidence matrix, where each “1” would represent an operation of a particular part on a particular machine and “0” otherwise. The block-diagonalization of the Machine-Part incidence matrix would be carried out with the help of a hybrid technique developed by combining GA with a Local Search Heuristic (LSH), while considering maximization of Grouping Efficacy (GE) as its performance factor. Once the cellular arrangement (machine groups and corresponding part families) is finalized, the user would be asked to provide information about the processing sequence of each part and the processing time for each operation of each part on each machine. Finally, taking this information into account the scheduling problem of the system would be solved as a typical JSSP and for that too a hybrid approach would be developed by combining GA with an LSH and considering minimization of Makespan as its performance factor. The ultimate output of the system would be a cellular arrangement showing machine groups and corresponding part families and a production schedule presented on a Gantt chart.

Since the methodology proposed here would not only provide solution for the Machine-Part grouping problem but solve the scheduling part of the problem as well. Therefore, it would definitely be more useful in practice and account for some of the major limitations of the available CMS design techniques such as “lack of using more production related information in the cell design" and "lack of ability to be practically implemented".

1.8 Research Objectives:

The main research objectives are listed and explained below. Each main objective has some intended novelties which are also listed and explained alongside their respective main objectives:

1. To develop a GA based hybrid approach (combination of GA with an LSH) for Machine-Part grouping - developing machine groups and corresponding part families - by considering maximization of GE as its performance factor, and validating the model by solving a number of benchmark problems from literature and comparing results with some of the known techniques. This technique has the following novelties:

   (i) Using integer type representation with the intention that information about the different groups of machines and their corresponding families of parts would be encoded in each chromosome. Another advantage of this type of representation is that it minimizes decoding effort. It is a rare practice as most of the researchers (for example Fernando & Mauricio [2002]) represent their solutions in the form of chromosomes that after decoding produce either machine cells or part families.

   (ii) Developing a repair algorithm that checks the legality of each solution that is randomly generated and the solutions those are resulted from the genetic operators (crossover, mutation and inversion). This would ensure that all the solutions in a population are legal and produce feasible solutions thus avoiding the penalty approach in which a new solution is randomly generated and placed in the population instead of an illegal solution which may disturb the natural evolution of GA.

   (iii) Developing an LSH that has the ability to increase the capability of traditional GA (that is using integer type representation, multi cut point crossover, gene to gene mutation and inversion) in terms of reaching the optimum in earlier generations. The LSH is to be placed inside the GA loop, so that the best solution of each generation is further improved and afterwards placed back into population so that it can participate in different genetic operators such as crossover, inversion and mutation giving it a chance to produce even better solutions. This is a novel approach as most of the LSH available in literature are subjected to the final outcome of GA.

All the relevant details are given in chapters to follow.

2. To develop a GA based hybrid approach (combination of GA with an LSH) for JSSP by considering minimization of Makespan as its performance factor, and validating the model by solving a number of benchmark problems from literature and comparing results with some of the known techniques. The intended novelties are as listed below:

(i) The use of integer type representation and arranging the entries in a two dimensional matrix, which is very rare in literature.

(ii) Devising a repair algorithm which ensures that all the chromosomes in population are legal and produce feasible solutions when decoded.

(iii) Here also, like objective 1, such an LSH is intended to be developed which can increase the efficiency of traditional GA (that is using integer type representation, multi cut point crossover and traditional swap mutation) in terms of reaching the optimum in earlier generations. Here also the LSH is to be placed inside the GA loop so that the best solution of each generation is further improved thus helping GA to reach the optimum in earlier generations.

All the relevant details are given in chapters to follow.

3. Linking the Machine-Part grouping part with the Job-Shop Scheduling part in such a way that the output of the Machine-Part grouping part can be served as an input to the Job-Shop Scheduling part. Finally, validating the combined approach by solving randomly generated or benchmark problems from literature. This is a novel approach as most of the research either focuses on cell design or cell scheduling. A combined approach of this kind is extremely rare to be found in literature.

1.9 Conceptual Approach:

To achieve the objectives discussed earlier the conceptual approach adopted is as shown in Figure 1.4.

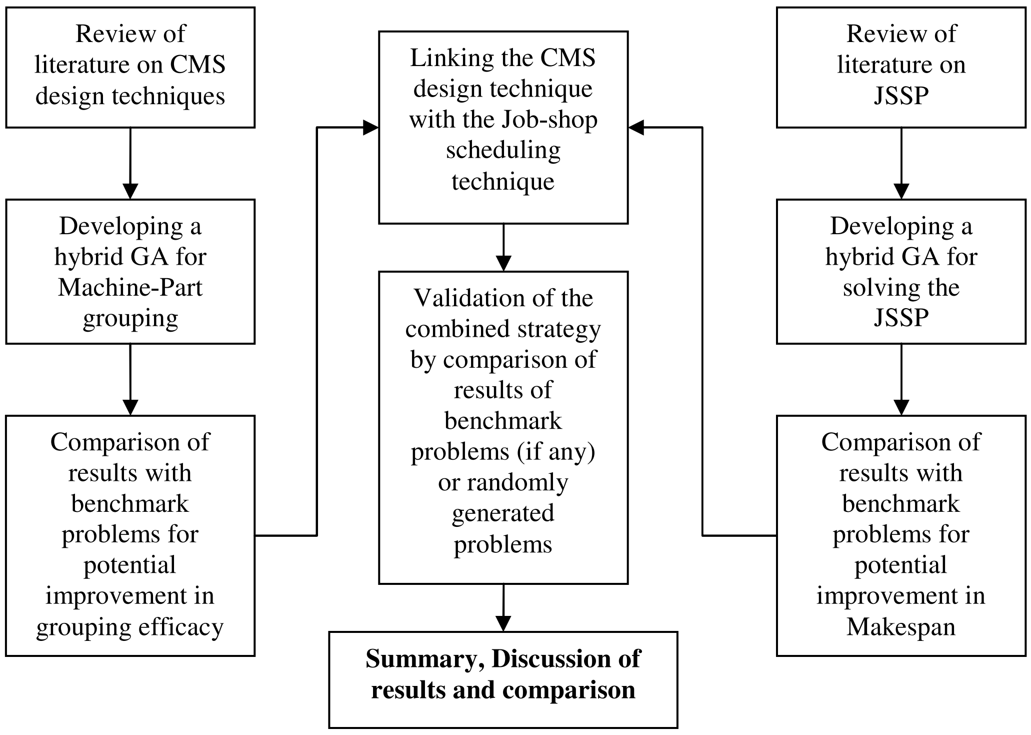

Review of literature on CMS design techniques
Developing a hybrid GA for Machine-Part grouping
Comparison of results with benchmark problems for potential improvement in grouping efficacy
Linking the CMS design technique with the Job-shop scheduling technique
Validation of the combined strategy by comparison of results of benchmark problems (if any) or randomly generated problems
Summary, Discussion of results and comparison
Review of literature on JSSP
Developing a hybrid GA for solving the JSSP
Comparison of results with benchmark problems for potential improvement in Makespan

Figure 1.4: Conceptual Approach for the Proposed Research

Figure 1.4 shows that the main parts of the approach are the development of hybrid GA based approaches that combine GA with LSH, both for the Machine-Part grouping and Job-Shop scheduling. Also, the next important aspect of the conceptual model is the linking of the two models together so that the output of Machine-Part grouping model can be served as an input to the Job-Shop scheduling model. For the hybrid GA in Machine-Part grouping part a simple integer based representation, with multi cut point crossover, traditional gene to gene mutation and inversion is applied. In order to measure the performance of a solution GE is used due to its inbuilt ability to maximize the utilization of a machine inside its cell and minimize the total number of intercellular moves. Also the capability of GE, as far as differentiating between the ill-structured and well-structured Machine-Part incidence matrices is concerned, is very high. Further, in case of GE no weight factor is required (Fernando and Mauricio [2002]). The LSH is placed within the GA loop so that the best solution of each generation is subjected to it and after being locally improved it is placed back into the population. The process is carried out for fixed number of generations and on termination the best result is saved.

Similarly, for the hybrid GA in Job-Shop scheduling part, also, a simple integer type, operation based representation is used. But it is worth mentioning that the way chromosomes are represented here i.e. in two dimensional arrays, is very rare to be found in literature. The traditional multi cut point crossover and swap mutation is applied here. In this part static scheduling condition is assumed, which definitely helps in facilitating the performance comparison of the proposed approach with the benchmark problems available in literature. Makespan is used as the performance measure, because it is a popular measure of performance and has been very frequently used by researchers. Here, also, LSH is incorporated inside the GA loop and the best solution of each generation is subjected to it.

1.10 Contributions:

This research provides an opportunity to have an in depth knowledge of the CMS design problem and the JSSP in general and the application of GA to both these problems in particular. Following are the major contributions of this research:

1. A hybrid GA is developed for solving the CMS design (Machine-Part grouping) problem by combining GA with an LSH. The technique, after being tested on a number of benchmark problems from literature, is found to be very effective as it has been found accurate and consistent irrespective of the problem size in comparison to all the other previously developed techniques. The effectiveness of the approach is due to the fact that a very effective LSH is present at the heart of the GA loop which locally improves the best solution of each generation and places it back into population so that it can take part in different genetic operators and produce even better solutions. The effectiveness of the LSH can be assessed by the fact that we have not used any specifically devised non-traditional genetic operators (crossover, mutation and inversion) but even

then results are better and consistent in terms of accuracy. This shows that the LSH is having a major effect on the overall GA procedure.

2. Another, hybrid GA is developed for solving the JSSP and here, too, GA is combined with an LSH. The technique, after being tested on a reasonable number of benchmark problems, has been found reasonably accurate and time efficient, as well. Here, also, the effectiveness of the technique is due to the presence of an effective LSH inside the GA loop which has the ability to produce accurate results even in combination with the traditional crossover (two cut point crossover) and mutation (swap mutation) strategies.
3. The two hybrid GAs (for Machine-Part grouping problem & JSSP) developed are linked together in such a manner that the output of the Machine-Part grouping (cellular arrangement) is served as an input to the Job-Shop scheduling part. The combination, which is extremely rare to be found in literature, has given a solution methodology that not only caters for the grouping of machines and corresponding part families but also provides solution for the system scheduling. This is indeed a distinguishing feature of this technique as it definitely accounts for some of the major limitations of the current CMS design techniques such as "lack of using more production related information in the cell design" and "lack of ability to be practically implemented".

1.11 Organization of Thesis:

This thesis report consists of eight chapters. Chapter 1 provides an introduction to this research, description of the research problem, research objectives, the conceptual approach to handle the research problem and the contributions of this research to the existing knowledge. Chapter 2 mainly focuses on the literature survey regarding different techniques of Artificial Intelligence (AI) in general and GA in particular as it has been frequently applied during this research. In Chapter 3 a comprehensive literature review regarding different techniques of CMS design is presented. Since during this research, not only the Machine-Part grouping problem is handled but solution for the system scheduling is provided as well, therefore a reasonable literature review regarding the Job-Shop Scheduling Problem (JSSP) is also carried out and presented in Chapter 4. Chapter 5 mainly describes the hybrid methodology developed to solve the cell design problem which is verified and validated through a number of benchmark problems from literature. Chapter 6 describes in detail a hybrid approach developed to solve the JSSP and also its combination with the approach developed for cell design (Machine-Part grouping). All the details about verification and validation of the hybrid tools, for Machine-Part grouping and Job-Shop scheduling, and further discussion regarding results of the problems is given in Chapter 7. Chapter 8 describes the main outcome (conclusion) of this research and also provides directions for future research in this area.

1.12 Summary:

This chapter has briefly given a background to the CMS design problem and the particular problem to be tackled during this research. The objectives expected to be achieved from this research have been described. A conceptual approach for solving the research problem has also been outlined that mainly consists of the development of hybrid GA based approaches both for the Machine-Part grouping and Job-Shop Scheduling parts, separately, and finally combining them together. It also describes the main contributions of this research to the area of CMS design. Finally the chapter discusses organization of the thesis.

CHAPTER 2

LITERATURE REVIEW- GENETIC ALGORITHMS

2.1 Introduction:

"Artificial Intelligence (AI) techniques have increased in terms of application in most of the fields in general and in the area of manufacturing system design and operations scheduling in particular" [Noor (2007)]. Therefore, it is very necessary to review the basics of AI, generally, and Genetic Algorithms (GA), more specifically as GA has been applied during this research. This chapter identifies the different AI tools, whereas GA has been discussed in detail.

2.2 Artificial Intelligence (AI):

McCarthy [1960] described AI as a subfield of computer science integrating the biological and computer intelligence. On the other hand, AI has been defined by Rich and Knight [1991] as the study of how to make computers do those things which at the moment are being tackled by people, in a better way. A more comprehensive definition of AI is given by Souri [2003], according to which it is the capability of a device or machine to perform functions which are normally associated with the intelligence of human beings.

The different branches of AI include Expert Systems (ES), Fuzzy Logic (FL), Artificial Neural Networks (ANN), Hybrid Systems and Genetic Algorithms (GA). Since GA has been more frequently utilized during this research, therefore, it has been further elaborated in detail, below.

2.3 Genetic Algorithms (GA)

GA is AI methodology that is inspired by the evolution theory of Darwin. In comparatively simpler words it can be said that in GA an evolutionary process solves problems and the final result is the best (fittest) solution (survivor) or in other words, it can be said that a solution is evolved. A brief description of the natural evolution process is discussed below which would help in thoroughly understanding GA.

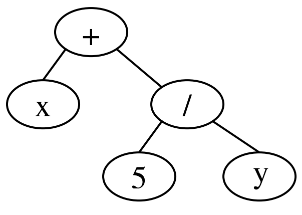
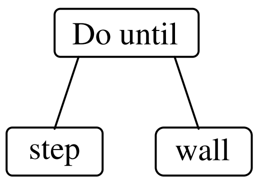
In nature all living organisms basically consist of cells. Every cell consists of a set of chromosomes. Each chromosome, in turn, is a string of DNA and serves as a model for the whole organism. A chromosome is basically a collection of genes, where each gene can be defined as a block of DNA and encodes a particular protein. In other words it can be said that each gene encodes a trait, for instance, the colour of eyes. The possibilities of different trait settings could be black, brown or blue. These settings are known as alleles. Every gene has a particular position in a chromosome and that position is termed as locus. As complete set of genes is called a chromosome, like wise a genome consists of a complete set of chromosomes. Whereas, a genotype is specified by a particular set of genes. It is the genotype that is mainly responsible for the after birth developments of the organism's phenotype, the different characteristics (mental & physical) for example colour of the eyes, level of intelligence, etc. Reproduction is the process during which new chromosomes are created. The first thing that occurs and is very important in the creation of new chromosomes is called crossover or recombination. During crossover genes from the parent chromosomes recombine and create new chromosomes. The other important operator that takes place during reproduction is called mutation. During mutation, basically, a small change is incorporated in the elements of DNA. Obitko [1998] observes that errors in copying genes from parents result in these changes. Survival is the measure of fitness of an organism.

Initially, Holland [1975] developed a methodology for GA that consists of a sequence of steps which are followed to move from one generation to another. In each generation the operators such as mutation and crossover are used for reproducing new chromosomes. Each chromosome’s performance or suitability is measured by some fitness value. This fitness value of a chromosome serves as a basis for its selection into the next generation.

Crossover is an operator during which two different chromosomes exchange their parts and hence develop offspring. Whereas, mutation is an operator during which a randomly selected gene of a chromosome is changed. As already discussed that fitness value is the basis on which the selection of the chromosome in the next generation depends, therefore, it can be said that a fitter chromosome has more chances of getting selected in the next generation. This fitness based selection ensures that the fittest chromosomes survive through generations whereas the least fit becomes extinct. The basic requirements of GA are: a fitness function that can measure fitness of a chromosome, an encoding criterion to encode a solution of a problem, defining the different constraints and criteria for the optimum value, and finally incorporation of suitable crossover and mutation operators. GA has the ability to perform efficiently in the evolution of an optimum solution, but the major difficulty in its implementation is the encoding of a problem solution, as improper encoding may lead to a complete change in the shape of a problem (Obitko [1998], Negnevitsky [2002]).

The first step in the implementation of GA is encoding i.e. the representation of a problem solution/ chromosome. Encoding is mainly dependent upon the problem to be solved. Some of the different encoding techniques are given in Table 2.1 (Noor [2009]). These have been used with some success [Obitko (1998)].

Table 2.1: Different Problem Encoding Techniques (Noor [2007])

S/No.	Encoding technique	Example
1	Binary encoding	Chromosome 1 :101100101100101011100101 Chromosome 2 : 111111100000110000011111
2	Permutation encoding	Chromosome 1: 8 9 7 4 6 2 3 5 1 Chromosome 2 : 2 3 1 4 8 5 6 7 9
3	Value encoding	Chromosome 1: 5.3243 1.2324 2.3293 0.4556 2.4545 Chromosome 2 : HDIERJFDJDLDFABDJEIFLFEGT Chromosome 3 : (right), (back), (left), (back), (forward)
4	Tree encoding	Chromosome A  (+ x (/ 5 y)) Chromosome B  (do until step wall)

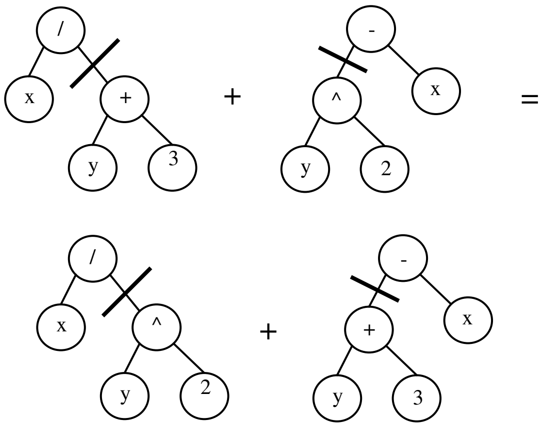
After encoding and the random generation of an initial population the next step in GA is the selection of chromosomes which would take part as parents in crossover. The main problem is how to carryout this selection. As per Darwin's evolution theory, the fittest chromosomes survive through generations and are most likely to take part in crossover and create offspring. "Some of the well known techniques of selection are tournament selection, roulette wheel selection, steady state selection and rank selection" (Noor [2007]). After the selection of chromosomes for crossover, the next task is to decide how to carryout the process of crossover so that genes from two parents can be recombined and children are created. The most common way of doing this is by random selection and then exchange of genes, as shown in example, below.

('|' is the crossover point):
Chromosome A=10011 | 10101110111
Chromosome B=**11001 | 01010011010**
Child A=10011 | **01010011010**
Child B=**11001** | 10101110111

Crossover can be carried out in a number of ways. Some of them are shown in Table 2.2. The type of crossover to be chosen mainly depends upon the type of encoding being used and therefore it can be sometimes quite complicated. A suitable selection of the type of crossover for a particular problem can definitely improve the GA's performance.

Table 2.2: Crossover Techniques for Various Encoding Types (Noor [2007])

S/No.	Encoding technique	Crossover technique	Example
1	Binary	1. single cut point crossover	**1100/1011**+1101/1111= **1100/**1111 + 1101**/1011**
		2. Double cut point crossover	**11/00/1011** + 11/01/1111 = **11**01**1011** +11**00**1111
		3. Uniform crossover	**11001011** + 11011101 =**11**011**111** + 1**1001**0**01**
		4. Arithmetic crossover	**0100110** + 0100101 = **11**00110 + **01**01111 **(the first three digits of each parent are added together for child 1 and the remaining digits are added for the child 2)**

2	Permutation encoding	Single cut point	**(1 2 3 4 5 6 /7 8 9) + (4 5 3 6 8 9/ 7 2 1) = (1 2 3 4 5 6** 8 9 7) **+**4 5 3 6 8 9 **7 1 2**
3	Real value encoding	Same as in binary values	**(1.29 5.68/ 2.86 4.11 5.55)+**(3.23 4.45/ 6.13 5.67 2.98) **= (1.29 5.68** 6.13 5.67 2.98) **+**(3.23 4.45 **2**.86 4.11 5.55)**--single point**
4	Tree encoding	Exchange	

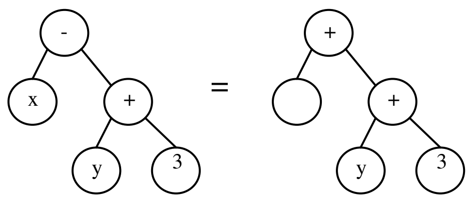
Once the issue of crossover is resolved, the next step is mutation. “The main aim of carrying out mutation is to induce a certain level of diversity into population so that GA can be prevented from getting trapped into a local optimum” (Obitko [1998]). It has been already mentioned, previously, that during mutation a slight change is incorporated in the genetic structure of a chromosome. An offspring, resulted from crossover, is randomly changed by a mutation operator. For binary encoding, mutation can be carried out as shown in example below.

(Bits selected for mutation are shown in bold)
Chromosome A=110**0**111000010010
Chromosome B=1101101**1**11110110
Mutated chromosome A=110**1**111000010010
Mutated chromosome B=1101100**0**111110110

Like crossover the decision of how to perform mutation, also, depends upon the type of encoding being used. The different mutation techniques, for different types of encoding, are shown in Table 2.3.

Table 2.3: Different Mutation Techniques for Various Encoding Types
(Noor [2007])

S/No.	Encoding type	Mutation technique	Example
1	Binary	Bit inversion	11001001 => 10001001

2	Permutation	Change in order	(1 **2** 3 4 5 6 **8** 9 7) => (1 **8** 3 4 5 6 **2** 9 7)
3	Real value	Addition of a small number	(1.29 2.86 5.68 **4.11** 5.55) => (1.29 2.86 5.68 **4.22** 5.55)
4	Tree	Change in operator	

"Crossover and mutation rates are the two basic parameters of GA" (Obitko [1998]). The crossover rate determines the number of times crossover of chromosomes will be carried out in one generation. The range for selection of crossover rate is from 0% to 100%. If the crossover rate is 0% it means that chromosomes in the next generation will be the exact copies of chromosomes in the current generation. On the other hand if it is 100% then every chromosome in the population of next generation will be the result of crossover between any two chromosomes of the current generation. Crossover is carried out in the hope that children, created during the process, would contain good parts of their parents and consequently perform better as compared to them. Each selection criteria is so designed that during selection some part of the population in the current generation do get selected in the next generation.

Similarly, mutation rate means how many genes in a population in one generation would get mutated. Here also the range could be from 0% to 100%. If the mutation rate is 0% then it means none of the genes would get selected. But, if it is 100% then it means all the genes in a population of a generation would get mutated. As indicated earlier, mutation is an operator that creates a certain level of diversity in a population and hence GA is prevented from getting trapped into local optimum. Therefore "selection of mutation rate is a delicate decision" (Noor [2007]). Too high a mutation rate would convert GA into a kind of random search and the characteristic of evolution is lost. Also, if the mutation rate is too low then there would be a tendency of GA converging on to a local optimum. In addition to two basic parameters i.e. crossover and mutation, there are some additional parameters of GA. One particularly important additional parameter of GA is the population size. Population size means total number of chromosomes in a population, in one generation. Population size is important because it provides GA the searching space in which search for the optimum solution is carried out. If there are too few chromosomes in a population then it means GA is given a smaller searching space and this would limit the GA's searching ability and there would be every likelihood of GA getting trapped on a local optimum. On the other hand, if there are too many chromosomes, then it means GA is provided with a larger searching space which would definitely slow it down by increasing its computational effort. Therefore, selecting a reasonable population size is again a delicate matter. According to Obitko [1998] it is not useful to use a very large population size because it does not solve the problem quickly as compared to a moderate sized population.

2.4 Summary

In this chapter a brief introduction to GA and its different parameters is presented. As far as its application to different fields in general and to the areas of CM and Job-Shop Scheduling in particular is concerned that is presented in detail in Chapter 3 and Chapter 4, respectively.

GA is a direct inspiration of the process of natural reproduction which consists of the processes of crossover, mutation and selection of chromosomes from one generation to another. Whenever, a problem is solved with the help of GA, the first hurdle that needs to be tackled is representation/ encoding. Since there are a number of encoding techniques available, therefore a suitable technique must be selected. The next in line is to select proper crossover and mutation techniques which should be in accordance to the encoding technique, already selected. Another important thing is to develop an appropriate fitness function which would evaluate the fitness of all the chromosomes in a population. These fitness values are responsible for the selection or rejection of a chromosome in the next generation as the fittest chromosomes have better chances of survival into the next generation as compared to their weaker counterparts. The mutation and crossover rates, number of generations and population size are the important GA parameters. These parameters are important because they determine the quality of the solution.

CHAPTER 3

LITERATURE REVIEW CELLULAR MANUFACTURING SYSTEM DESIGN TECHNIQUES

3.1 Introduction:

GT is in fact a concept of manufacturing that takes into account the similarities in design and processing requirements of different parts while grouping them together in to families (Irani [1999]). Mitrofanov [1966] and Burbidge [1975] gave the initial concept of GT and Burbidge [1975]. GT can be defined as, "a method of manufacturing piece parts by the classification of these parts into groups and subsequently applying to each group similar technological operations" (Mitrofanov [1966]). Another definition of GT is, "the realization that many problems are similar and by grouping them a single solution can be found to a set of problems, thus saving time and effort" (Shunk [1978]). This definition gives a comparatively wider understanding of GT. However, a more general definition of GT is "it is a manufacturing philosophy which identifies and exploits the underlying proximity of parts and manufacturing processes" (Ham et al. [1985]).

3.2 Classification of Cellular Manufacturing System (CMS) Design Techniques:

Basically, CM is a concept based on the application of GT to the reconfiguration of manufacturing systems. "CM is a hybrid system linking the advantages of both job shops (flexibility in producing a wide variety of products) and flow lines (efficient flow and high production rate)" (Mungawattana [2000]). In CM the machines are normally close to each other and involved in the manufacturing of a specific part family. This particular arrangement facilitates the system to have an efficient flow of material and a comparatively higher rate of production like an FL. By using general-purpose machines the CMS can be easily reconfigured to facilitate the production of new product designs and product demand with a comparatively smaller amount of effort both in terms of cost and time. This makes the manufacturing system more flexible in terms of producing different variety of products.

"CM is a manufacturing system that can produce medium-volume/medium-variety part types more economically than other types of manufacturing systems" (Black J. [1983]). The different categories, of the numerous GT algorithms that can be repeatedly found in literature, are as follows:

a) Classification and coding
b) Array-Based clustering
c) Graph Partitioning
d) Similarity Coefficient
e) Mathematical Programming
f) Artificial Intelligence (AI) Based
g) Heuristic based approaches

These available techniques have been reviewed in detail by Joines et al. [1996]. According to them (Joines et al [1996]) these techniques can be divided into two main classes as design oriented techniques and production oriented techniques. The criteria of grouping parts in the design oriented techniques is based on similarity in design features, whereas, in production oriented techniques grouping is carried out on the basis of similar processing requirements. Joines et al. [1996] presented the division of CMS design techniques as shown in Figure 3.1:

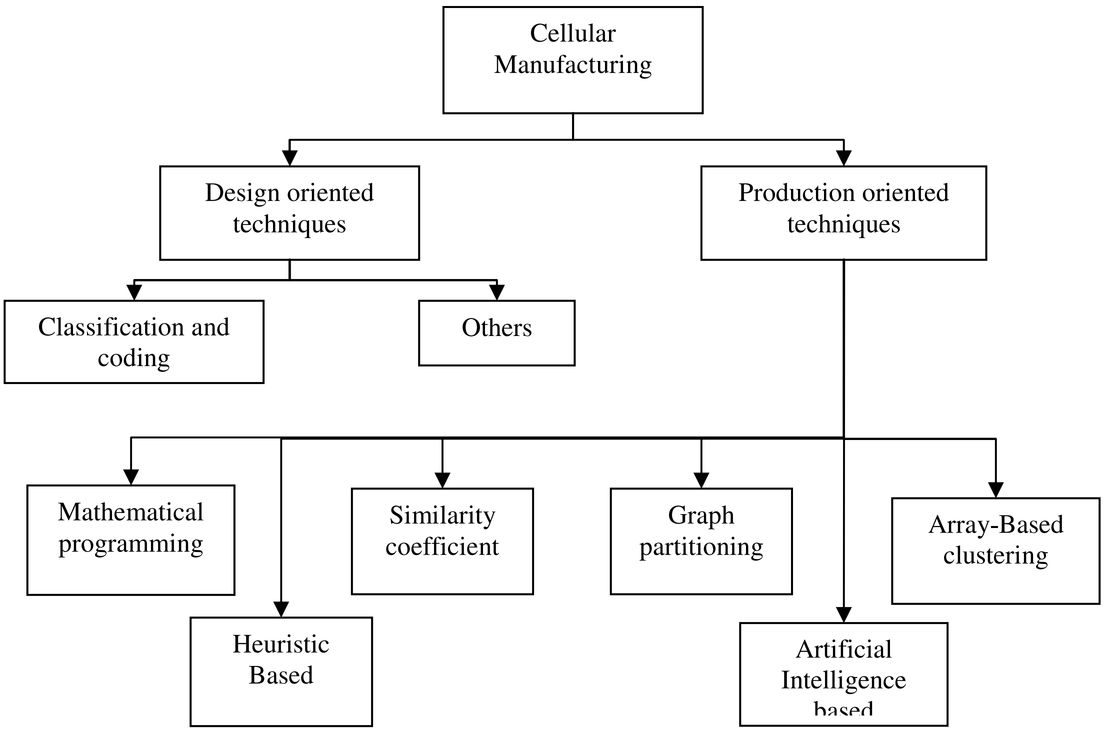


Figure 3.1: Classification of machine-part grouping techniques

a) Classification and Coding Analysis:

This approach uses a coding system to assign numerical weights to parts characteristics and identifies part families using some classification scheme. The classification and coding of parts is a very complex job and takes a lot of time. Perhaps due to the design orientation and proprietary nature of most coding systems, they are not popularly known in the research literature for GT or cellular manufacturing. Like other grouping techniques this method also tries to group parts into families on the basis of similarities in design and manufacturing attributes. This concept of grouping similar parts into families using design features was initially introduced by Mitrofanov [1966] and Opitz et al. [1969]. Different attributes of a part like shape, dimensions, size of hole, size of gear tooth etc, are all encoded in the form of a code number. The different design manufacturing attributes of parts are summarized in Table 3.1:

Table 3.1: Design and Manufacturing Attributes of Parts.

S/No.	Design attributes	Manufacturing attributes
1	Basic external shape	Major processes
2	Basic internal shape	Minor processes
3	Rotational or prismatic shape	Process plan

4	Length-to-diameter ratio (rotational)	Major dimensions
5	Aspect ratio (prismatic)	Surface finish
6	Material type	Machine tool
7	Part function	Production cycle time
8	Major dimensions	Batch size
9	Minor dimensions	Annual production
10	Tolerances	Fixtures required
11	Surface finish	cutting tools

A code number completely represents the size, dimensions, shape etc of a part. Parts carrying same code numbers can be grouped into families as they would be having similar attributes and hence similar processing requirements. Machines are grouped according to the processing requirements of each part family.

Classification and coding based systems were the primary tool of GT in the 1960s and 1970s. A number of classification and coding systems are reported in literature. According to Opitz, H., [1970] "classification and coding techniques have been used in practice but using features, based on shape and then group parts accordingly is a very labor intensive job and in order to solve this problem the idea of weighted codes is useful". Kaparthi and Suresh [1990] suggested that the lack of popularity of classification and coding based systems might be because they are labor intensive, and thus they suggested the automation of the coding part of the method. Ham et al. [1985] provided a complete survey of the different classification and coding techniques. Kusiak [1987] argued that the unpopularity of coding based techniques could be due to the expense and difficulty involved in implementing coding and classification systems; however, Hyer and Wemmerlov [1989] and Levuhis [1978] viewed it otherwise. For example, according to a survey conducted by Hyer and Wemmerlov [1989] showed that 62% of companies surveyed indicated that they used classification and coding systems; therefore one of the reasons for the slow development of coding based algorithms could be finding groups with the weighted codes. Offodile

[1991] came up with the idea of developing a framework by which the weighted codes can be converted into a similarity measure and argued that this could alleviate the problem.

Billo [1999] identified and described several organizing principles (Operational Constructors, Meta Models, Dynamic Binding and Associative Naming) and argued that if software is developed using these principles then it would be more user friendly. Liao [2001] came up with the idea of using Fuzzy Logic in devising a classification and coding based system for the formation of part families. The authors argued that traditionally all features of a part were considered as crisp, whereas realistically some should be fuzzy. Barton and Love [2005] developed a novel system that was based on the already existing coding system CAMAC which was originally developed by Love and George [1985]. "The ability to find designs using a detail drawing rather than textual descriptions is a significant achievement in itself" Barton and Love [2005]. They further argued that if certain way could be found to find parts from simple sketches then this would be more useful and effective in practice.

There are a number of coding packages available commercially. However, none of the systems has universally been adopted because of the fact that some require manual coding which is extremely slow (not more than 100 parts per day) and would take years to code the complete database of even a small company having few thousand parts. Automatic coding models using three-dimensional models would eliminate this problem but even then the overall approach remains non-simultaneous. This (non-simultaneous way of grouping parts into families and machines into corresponding groups) is a major weakness of the classification and coding based techniques.

b) Array Based Clustering:

Array based clustering is one of the simplest as compared to other production oriented techniques used to form machine cells and corresponding part families, simultaneously. The system is initially arranged in the form of a 0-1 Machine-Part incidence matrix that consists of entries in the form of either 1s or 0s. 1 means the operation of a part *j* on machine *i* (a_{ij}=1), and 0 means otherwise. Once the system is arranged in 1-0 form then with the help of shifting columns and rows all or most of the 1s can be rearranged in the form of blocks along the diagonal of the matrix. Each block in the diagonal represents a family of parts and corresponding machines which are simultaneously formed, as shown in example presented in Tables 3.2 & 3.3:

Table 3.2: Incidence Matrix

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

Table 3.3: Block-diagonalized form

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

A detailed comparison of some of the well known array-based clustering techniques (Rank Order Clustering (ROC), Direct Clustering Analysis (DCA) and Bond Energy Analysis (BEA)) is given in Chu and Tsai, [1990].

ROC was initially proposed by King, [1980]. In ROC every row and column of the machine part incidence matrix is considered as string of binary numbers. The block-diagonalization of the matrix is achieved by rearranging the rows and columns in the decreasing order of the decimal values of their respective binary strings. Chanrasekharan and Rajagopalan, [1986] identified the limitations of ROC. They observed that "ROC is highly dependent upon the configuration of the input matrix and therefore can only be efficiently applied to well-structured Machine-Part incidence matrices". Despite its shortcomings, ROC is considered to be the most simple and popular algorithm for cell formation.

"BEA is a general clustering algorithm and can be applied to any nonnegative array of numbers" (McCormick et al. [1972]). According to this technique the interrelationship, between an element in an array and its neighboring four elements, is being exploited. The sum of the products of these adjoining elements creates bond energy. The total bond energy (TBE) for a particular row 'x' and column 'y' can be calculated as shown in (3.1):

$$TBE(x,y) = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij} * [\, a_{i,j-1} + a_{i,j+1} + a_{i-1,j} + a_{i+1,j}] \qquad (3.1)$$

Where:

$a_{0,j-1} = a_{m+1,j} = a_{i,0} = a_{i,n+1} = 0$

m = *the number of machines*

n = *the number of parts*

The BEA, in fact, maximizes the TBE over all the *n!m!* permutations (Joines et al. [1996]). The main weakness of this method is that it allows the formation of a final Machine-Part matrix that contains overlapping blocks which makes it difficult for the viewer to identify the natural machine groups and their corresponding part families. This becomes impossible when the problem size is large.

DCA was proposed by Chan and Milner, [1982]. According to their approach the initial matrix is rearranged by moving rows with the left most positive cell to the top and columns with the top most positive cell to the left. A positive cell is the one where $a_{ij}=1$. The main advantages of DCA are that unlike ROC it does not have any size limitations, converges to the final solution in comparatively lesser number of iterations. Before applying the algorithm, all the exceptional elements and bottleneck machines are removed after being marked. However, Wemmerlov, [1984] observes that DCA may not produce acceptable solutions as the diagonal is redirected in each iteration. Therefore he came up with a modified version of DCA.

After comparing the three array based clustering techniques it was concluded that BEA outperformed the other two methods (ROC & DCA) especially in the presence of exceptional elements (Chu and Tsai, [1990]).

The main problems with the clustering techniques are the presence of a large number of exceptional elements and bottleneck machines on which a large number of parts need to be processed. Therefore, "quality of the results given by clustering techniques is highly dependent upon the initial input i.e. the configuration of the zero-one Machine-Part incidence matrix. Therefore, these approaches loose their effectiveness as the problem size increases and the Machine-Part incidence matrix gets more and more ill-structured" (Tariq et al. [2007]). These are the reasons that these techniques are very rarely used these days. They have been over taken by some of the well known modern approaches of cell formation that utilize Simulated Annealing, GA, Artificial Neural Networks, Fuzzy Logic etc.

c) Graph Partitioning Approaches:

The first to implement the graph partitioning approach to the cell formation problem were Rajagopalan and Batra, [1975]. The graph they developed to solve the cell formation problem was basically a combination of nodes and arcs. Each node represented a machine, whereas, each arc indicated similarity among the machines. The partitioning of graph or, in other words, the formation of machine cells, was carried out by assembling cliques determined from the graph. The work of Rajagopalan and Batra, [1975], also, pointed out that normally the number of intercellular moves does not reflect the true cost of material handling. Especially in the case when an intercellular move is somewhere in the middle of the processing sequence of a part. In such a situation there will be two intercellular moves required instead of one.

"Normally, graph partitioning approaches treat the machines and/or parts as nodes and the processing of parts as arcs connecting the nodes" (Askin and Chu, [1990]). One thing that is considered as the major weakness of graph partitioning approach is its non-simultaneous grouping of parts into families and machines into respective cells. The approach developed by Kumar et al. [1986] was also based on graph partitioning during which they solved the Machine-Part grouping problem for a known number of cells while applying a constraint on the size of cell.

Other approaches that are actually the extension of the initial graph partitioning approach, include, the bipartite graph approach of King and Nakornchai, [1982], a heuristic and graph partitioning approach of Askin and Chu, [1990], the network flow approach of Vohra et al. [1990], and the Minimum Spanning Tree (MST) approach of Ng, [1992] & [1993], As mentioned earlier this approach is non-simultaneous and that is why either would be applied to the formation of part families followed by grouping of machines or otherwise. Also, being an analytical approach it cannot handle large size of problems especially those which have ill-structured Machine-Part incidence matrices. Due to these limitations graph partitioning approaches are not very popular and that is why limited number of researchers applied it to the cell formation problem. One recent example in which graph partitioning has been applied to the cell formation problem is Rebeiro [2009]. In this technique graph regarding the production system is generated first and then a coloring algorithm is activated to obtain the number cells as desired initially. Though the technique is claimed to be an effective one but the results presented show that it has been applied to comparatively smaller size of problems and in limited number of problems (3 out of a total of 15 problems) improvement could be observed.

d) Similarity Coefficient Approach:

In this approach first similarity measures among parts are determined by taking into consideration their features, designs and processing requirements and then formation of part families and their corresponding machine groups is carried out using these similarity measures (Irani [1999]).

The main steps to develop machine groups and part families with the help of similarity coefficient based approaches generally are, as described below:

i. The initial data is provided in the form of a 0-1 Machine-Part incidence matrix of order $M \times P$ where, M represents total number of machines and P represents total number of parts. Each element (a_{ij}) of the initial Machine-Part incidence matrix is either 1, if part j has an operation on machine i, or 0, if otherwise.
ii. By using the Machine-Part incidence matrix the similarity coefficient is calculated and the initial matrix is converted into a triangular form representing either machine-by-machine ($M \times M$) or part-by-part ($P \times P$) similarity matrix.

iii. Using the information available after step ii a tree/dendrogram is constructed which actually represents the hierarchy of similarity among all pairs of machines or parts.
iv. From the tree/dendrogram developed in step iii the corresponding machine groups or part families are formed while satisfying the constraints e.g. cell size etc.

Once the similarity coefficients between all the pairs of machines and parts are calculated the initial machine part incidence matrix can be converted into a final block-diagonal form showing groups of machines and corresponding part families by following steps iii and iv, described above. McAuley [1972] was the first to develop machine cells by using the Jaccard similarity coefficient. McAuley's work was applied to a number of real life problems by Carrie [1973]. He came up with the idea to apply similarity coefficient approach to form part families first. But it was later on observed by White [1980] that it would not be carrying any advantage if part families are formed first followed by the grouping of machines into cells or otherwise.

Since similarity coefficients have the ability to incorporate the manufacturing data other than the binary machine part incidence matrix, therefore a variety of such methods have been developed, some of the well known examples are: the average linkage method of Seifoddini and Wolfe [1986], the similarity coefficient approach of Seifoddini [1989] which was developed to handle the problem of improper machine assignment, and the production data based similarity coefficient approach of Gupta and Seifoddini [1990] that had the ability to incorporate a variety of production related information such as the Machine-Part incidence matrix, the actual sequence of operations, the average production volume of each part, the processing time of each operation of a part.

Mosier and Taube [1985] introduced two kinds of similarity coefficients. One is called the Additive Similarity Coefficient (ASC) and the other is called Multiplicative Similarity Coefficient (MSC). The ASC is actually the implementation of the conventional jaccard similarity coefficient in a weighted manner. Each part, in this case, is assigned a weight according to its importance. MSC, on the other hand, is more or less a correlation coefficient. The conventional Jaccard Similarity Coefficient was, also, modified by Seifoddini and Djassemi [1995] in order to accommodate the information related to production volume. The modified version of the conventional Jaccard Similarity Coefficient proved more effective both in terms of reducing the total number of intercellular moves and improving the process of scheduling. Geonwook et al [1998] developed a two-phase procedure for configuring a CMS. In phase-I a new similarity coefficient was developed that presented the option of alternative routings in case of machine failures, whereas, in phase-II a methodology, to consider scheduling and operational aspects of the cell design, was developed.

There are a number of other similarity coefficients being developed and used in solving the cell formation problem. Shafer and Rogers [1993b] carried out a comparative study on 16 different similarity coefficients. A number of comparatively recent studies carried out in the application of similarity coefficients to the CMS design problem, includes, Onwubolu and Mlilo [1998], Ben-Arieh and Sreenivasan [1999], Lozano et al. [1999], Baykasoglu and Gindy [2000], Samatova et al. [2001].

Yasuda and Yin [2001] reviewed two similarity coefficients; Jaccard Similarity and Commonality Score, and concluded that these similarity coefficients are inefficient to solve the cell formation problem. Yin and Yasuda [2005] conducted a comparative study and concluded that though a generalized similarity coefficient cannot be devised for the cell formation problem but at least it could be found that which similarity coefficient is comparatively appropriate for a particular situation. In their further study (Yin and Yasuda [2006]) the authors conducted a comprehensive review of similarity coefficient based approaches. In addition to presenting a detailed review of the similarity coefficient based techniques, a review of the previous reviews has also been presented. It has also been argued in this paper that similarity coefficient is the most flexible cell formation method.

Though similarity coefficients based methods offer greater flexibility in terms of incorporating production level data while solving the cell formation problem, there exists a number of deficiencies in the current similarity coefficient based methods. Most of the techniques developed so far either form part families first followed by machine groups or otherwise. Existence of simultaneous Machine-Part grouping approaches based on similarity coefficients is very rare in literature. Also, even with the non-simultaneous approach handling large size problems is not possible and that is why recent similarity coefficient based approaches utilize GA when it comes to solve large size cell formation problems.

e) Mathematical Programming Based Approaches:

Approaches based on Mathematical Programming actually solve the Machine-Part grouping problem as an optimization problem. Due to their ability to consider and incorporate a number of critical system design information, Mathematical Programming based approaches have been extensively used to solve the Machine-Part grouping problem. All the mathematical optimization approaches applied to the cell formation are either linear or nonlinear integer programming problems. Kusiak's, [1987 & 1988], and Boctor's, [1991], work has shown that these approaches have the ability to incorporate a lot of production related data, for example; processing sequence, routing flexibility, setup and processing time etc. Being an optimization technique, the objective; while clustering parts or machines; could be to maximize the total sum of similarities between each pair of machines/ parts. A review of the work of some of the researchers, who used mathematical programming techniques to formulate the cell formation problem, is presented in Table 3.4.

Table 3.4: Mathematical Programming Based Approaches

S/No.	Reference	Technique	Review
1	Purcheck [1974,1975]	Linear integer Programming	The author was the first to have applied the approach of mathematical modelling to the cell formation problem. The research shows that first the part families are formed and then machines are grouped according to the processing requirements of each part family which indicates that the approach carries out Machine-Part grouping sequentially rather than simultaneously.

2	Choobineh [1988]	Integer Programming	The methodology presented in this paper consisted of two stages. In the first stage part families were formed using clustering techniques with a proximity measure that took manufacturing operations and their sequence into consideration. In the second stage, also, an integer programming model was proposed to assign machines to different cells. The procedure itself reveals that it is not a simultaneous approach.
3	Shtub [1989]	Mathematical Programming	Here a mathematical model for the cell formation problem was developed as a generalized assignment problem.
4	Gunasighe & Lashkari [1989, 1989, 1990]	0-1 integer programming	The authors developed two separate 0-1 integer programming models. One for the part family formation and the second one for the grouping of machines into cells. The objective of their model was to minimize the cost involved in the total number of intercellular moves and the duplication of machines. The concept of routing flexibility was also considered during this research. Since separate models were developed for the part family formation and grouping of machines therefore this shows that the approach is not a simultaneous one and also finding solution, analytically, for the large scale problems would be difficult.
5	Srinivassan et al. [1990]	Generalised assignment approach	In this research the authors presented a generalised assignment approach for the cell formation problem and argued that it would perform better than the *p*-median model. The main feature of the approach was that the number of part families (number of cells) was not known as a *priori*. This was also a sequential approach of grouping parts into families first and then assigning machines to each cell afterwards.
6	Boctor[1991]	Mathematical Programming	The author presented an analytical optimization model for simultaneous Machine-Part grouping with the objective of minimizing the total number of intercellular moves. Since it was an analytical approach therefore it could not be applied to problems of large size. This made the author to recommend Simulated Annealing approach.
7	Joines et al.	Integer	Here, also, an integer programming

	[1996]	Programming	model was developed for the cell formation problem. But instead of solving problems analytically, the authors made use of GA which speaks itself about the limitations of pure mathematical based techniques.
8	Heragu & Chen[1997]	Mathematical Programming	The authors came up with an approach based on mathematical programming and claimed that they could find an optimal solution for a cell formation problem. The model consisted of three major aspects - routing flexibility, machine utilization and practical constraint. Though it was an efficient approach but could only solve problems of medium size.
9	Cheng et al. [1998]	Mathematical Programming	Here the problem of cell formation was initially formulated as the Travelling Salesman Problem (TSP) based on mathematical modelling. Since finding analytical solution is excessively time consuming therefore GA was used to find solution for the problems while using GE as the performance measure. This shows the limitations of pure mathematical techniques.
10	Chen & Heragu [1999]	Mathematical Programming	Here the authors improved their previous work (Heragu & Chen [1997]) and tried to over come some of its limitations (e.g its lack of ability to solve large scale problems). The methodology presented in this paper was based on the stepwise decomposition of the large scale cell formation problem.
11	Won [2000]	*p*-median based	During this research the cell formation problem was formulated as *p*-median based mathematical model using a comparatively new similarity measure between pairs of machines. Since similarity measure is used between pairs of machines this means that first machine groups would be formed and then parts would be assigned afterwards. This shows that the technique is not grouping parts and machines simultaneously.
12	Akturk & Turkcan [2000]	Mathematical Programming	In this paper the authors have integrated the cell formation problem with the cell layout problem while carrying out the Machine-Part grouping simultaneously.
13	Plaquin & Pierreval	Mathematical Programming	An approach based on mathematical programming and taking into account

	[2000]		some specific constraints, e.g some machines could stay together if they shared a common resource and some causing interference would be placed separately, was presented in this paper. The authors devised an evolutionary approach to solve the cell formation problems which speaks itself about the limitation of using mathematical modelling and solving the cell formation problems analytically.
14	Zhao & Wu [2000]	Mathematical Programming	In this research the authors have developed a multi-objective optimization model for the cell formation problem. The multiple objectives they considered, included minimization of the cost involved in intercell and intracell movements, minimization of the variations in cell load, and minimization of the total number of exceptional elements. The cell formation problem was actually solved as machine grouping problem and parts were assigned to each group considering their processing sequence and production volume. Since the problem was formulated as multi-objective, therefore analytical solution for even the problems of medium size was not possible and that is why the authors used GA. The computational experience presented in the paper is limited and even in the presence of GA it can only sove problems of medium size only.
15	Caux et al. [2000]	Mathematical Programming	Here the authors developed an optimization model for the cell formation problem while considering alternative process plans and the machine capacity constraint with the objective of minimizing the intercellular moves. Since the model was multi-objective therefore analytical solutions could not be determined and that is why the author proposed an SA based methodology and combined it with a branch-and-bound technique for the routing selection. The approach presented is a logical one but the computational experience presented in the paper is very limited which may not be enough to justify its effectiveness when applied to large size practical

			problems both in terms of accuracy and computational time.

Some of the comparatively recent research studies that utilized mathematical programming to formulate the cell formation problem includes Onwubolu and Mutingi [2001], Uddin and Shanker [2002], Nsakanda et al. [2005], Fantahun et al. [2006], Geonwook and Herman [2006], Tariq et al. [2006], Tariq et al. [2009].

Onwubolu and Mutingi [2001] formulated the cell formation problem by taking into account the cell-load variation. The work of Uddin and Shanker [2002] was based upon integer programming with the consideration of routing flexibility. Nsakanda et al. [2005] developed a mathematical approach that considered a number of many manufacturing parameters including the part demands, multiple routings, operations sequence, multiple process plans, and machine capacities. The drawback of their methodology was that it could never guarantee optimality.

Fantahun et al. [2006] formulated the cell formation problem as a mathematical optimization problem while considering a number of information such as; routing flexibility, operation sequence, machine duplication, machine capacity, cell load balancing, and different type of costs like operation, tool consumption, setup and subcontracting cost. Since the model was multi-objective and was also multi-constrained therefore to solve the cell formation problem analytically with the help of this model was a challenging task. Therefore, the authors proposed a GA based heuristic approach to solve the cell formation problem.

Geonwook and Herman [2006] presented a two-phase mathematical approach for the cell formation problem. In the first phase formation of part families was carried out whereas in the second phase in addition to the formation of corresponding machine groups, operational issues were also handled. Since the problem was multi-objective and multi constrained therefore finding solutions analytically for medium and large size problems in polynomial time would not be possible and that is why the authors proposed a GA based methodology. The approach seems to be an effective one but computational experience presented in the paper is very limited and a comparatively smaller size problem is solved.

Tariq et al. [2006] formulated a mathematical optimization model for the cell formation problem with the objective of maximizing GE: which consequently minimizes the total number of intercellular moves and maximizes a machine's utilization into its respective cell. Initially they applied their model to small and medium sized problems but in their later work (Tariq et al. [2009]) a comprehensive computational experience of the same model is presented. They also suggested a hybrid GA based methodology to solve the cell formation problem.

Though the mathematical optimization approaches are very attractive in terms of incorporating more production level data into cell formation models but to find analytical solution of medium and large size cell formation problems in polynomial time is not possible which is a major limitation of mathematical optimization based techniques. "Obtaining optimal solution for the mathematical programming approaches can be infeasible due to combinatorial complexity of the CMS design problem" (Mungawattana [2000]). Most of the recent researchers have, therefore, opted for either SA or GA based methodologies rather than pure mathematical approaches.

f) Artificial Intelligence (AI) Based Approaches:

The application of AI-based approaches to the problem of CMS design has increased in recent years. Though it may seem as if these approaches are patterned on the same structure as that of the conventional array based clustering techniques, but still they are different as they have the ability to incorporate AI in their algorithms. Another feature is their searching ability through which they can find out optimum or near to optimum solutions with comparatively little computational effort. They also have the ability to handle multiple objectives and related constraints, quite conveniently. Some of the well known AI techniques are listed as follows:

i. Artificial Neural Networks (ANN)
ii. Fuzzy Logic (FL)

iii. Simulated Annealing (SA) & Genetic Algorithms (GA)

A brief description of these techniques and the related research work, being carried out so far, is given below:

i. Artificial Neural Network (ANN):

ANN are the network models developed for computations. They contain simple units that carryout processing and communicate with each other by exchanging signals through a number of connections where each connection carries a weight factor. ANN is actually the inspiration of human brain.

Like a human brain, ANN is also a network containing a number of processing units known as artificial neurons. These processing units are connected with each other through weighted connections. Each artificial neuron transmits the incoming signal to other units through their outgoing weighted connections. The information transferred among processing units is stored in these weights in the form of specific values. This storage of information enables these networks to have the ability of learning from experience and memorizing the relationships among different data.

ANN has been successfully applied to the area of manufacturing. Because of some of its unique capabilities ANN is highly suitable for application to the area of CMS design. ANN's use for CM applications is justified by the fact that they can learn from experience, recognize patterns, and generalize the knowledge they obtain. Another advantage is their ability to handle incomplete data which is highly useful when it comes to real world applications. Due to these advantages, ANN has been extensive used in the area of GT in recent years.

ANN can be trained in two ways; through supervised learning or unsupervised learning. A single layer perceptron and multi-layer perceptron are the two important architectures of supervised learning (Noor [2007]). "A perceptron can represent a linearly separable function, and hence it can learn only OR and AND operations but not the Exclusive-

OR (XOR) which is not linear in nature" (Noor [2007]). This limitation of perceptron can be overcome by training a multilayer ANN with back-error propagation (Negnavitskey [2002]). But even, "back-error propagation is not the exact emulation of human brain and hence the multi-layer perceptron does not have associative memory characteristics like a human brain" (Noor [2007]). Jain and Meeran [1999] and Meeran [2003] reports that ANN sometimes perform poorly due to lack of data or the trajectory dependent training algorithms which cannot map the complex data precisely.

The basic requirement of supervised learning techniques is to have some knowledge in advance about the number of cells to be formed and a set of training data with known output. The fact of the matter is that the availability of this kind of knowledge in advance is very hard in case of Machine-Part grouping problem. The reason is that, generally, no information about the exact number of groups to be formed is known a *priori*. But still, a number of researchers have applied supervised learning technique to solve the cell formation problem. A three-layered feed-forward network with back-propagation (supervised learning) for part grouping in CMS was proposed by Kao and Moon [1991]. They overcame this difficulty of training data by selecting some seed parts as representatives of the part families to be formed. Some of the earlier research works in which supervised learning is used for part/ machine grouping, include Kao and Moon [1990], Moon [1990], Moon [1990], Moon and Chi [1992], Moon and Roy [1992].

Unsupervised learning is suitable for the initial cell formation. Once the initial formation is over then supervised learning can be applied and trained to refine the formation and add new parts to cells. That is why majority of neural networks that have been developed for parts/ machines grouping used unsupervised learning methods. Some of the unsupervised learning models developed over the years include, Self Organizing Feature Maps (SOFM) - Kohonen [1982], Adaptive Resonance Theory (ART) - Carpenter and Grossberg [1987], the modified Hebbian learning algorithm of Malave and Ramchandran [1991], and the Fuzzy Adaptive Resonance Theory (Fuzzy ART) - Carpenter and Grossberg [1991].

"SOFM performs a dimension reduction of the input patterns to the two-dimensional space while maintaining the topological relations between the elements" (Kiang et al. [1995]). SOFM operates in a similar manner as the competitive learning model. SOFM has been found efficient when applied to the problem of part grouping.

Like any competitive learning model the ART tries to group parts automatically. In fact the output layer's neurons directly represent part families. The vigilance threshold enables the ART network model to control the similarity among parts of a part family. Another advantage of this network is that it does not need the information, about the number of part families to be developed, in advance. Some of the researchers who used ART for part/ machine grouping and investigated some of its drawbacks include, Kaparthi and Suresh [1992], Kaparthi et al. [1993], Dagli and Huggahalli [1995], Chen and Cheng [1995].

Fuzzy ART, Carpenter and Grossberg [1991], is an improved form of ART carrying the concepts of Fuzzy logic. A basic difference between ART network and Fuzzy ART is that both non-binary and binary data can be handled by Fuzzy ART. Suresh and Kaparthi [1994] used a Fuzzy ART network to develop part families using the Machine-Part incidence matrix. They also compared the performance of Fuzzy ART with conventional ART as well as ROC [King, (1980)] and found Fuzzy ART superior to these techniques. Because of the promising results achieved, Fuzzy ART is still considered to be an open area for research. The work of Suresh and Park [2003] is basically an extension of the conventional Fuzzy ART and which allows the consideration of operation sequence while clustering the parts. Peker and Kara's [2004] work is actually an investigation on parameter setting for Fuzzy ART networks.

Although a lot of research is being carried out in improving the ANN algorithms, applied to the problem of cell formation, however little work has been carried out to handle the problem of bottleneck machines. Also, some additional constraints, for example load balancing, capacity constraints, or part demands etc., need to be considered, at the CMS design stage, as well.

ii. Fuzzy Logic (FL):

FL was invented by Lofty Zadeh, a professor at the University of California at Berkley, in 1965. The motivation for the development of FL was that, “imprecisely defined sets or classes play an important role in human thinking, particularly in the domain of pattern recognition and communication of information” (Zadeh [1965]). FL gives designers the abilities; to define relationships in less than exact terms, to handle the simulation of real world judgements with ease, to cope with nonlinearity with the same ease as linear relationships, to be able to simulate natural language intercourse, and finally to have the ability to handle real life problems with ease and more flexibility. FL is an interesting AI tool as it gives us a simplified approach to reach the definite solution of a problem while utilizing imprecise or ambiguous input information.

Fuzziness has been handled in many ways in case of cell formation problems. A review of some of the earlier research works in this area is presented in Table 3.5.

Table 3.5: FL Based Techniques

S/No.	Reference	Technique	Review
1	Xu & Wang [1989]	FL	In this research the authors used fuzzy mathematics to handle the issue of uncertainty/imprecision while calculating similarity between parts. Here a dynamic part family formation procedure has been presented which assigns new parts to an already existing part family by using the principle fuzzy pattern recognition. The weakness of the approach could be its non-simultaneous approach towards Machine-Part grouping.
2	Chu & Hayya [1991]	Fuzzy C-mean algorithm	The authors were the first to have applied the Fuzzy C-mean Algorithm, initially developed by Bezdek [1991], to the cell formation problem. Their algorithm resulted in two matrices one for the identification of the part families and second for the identification of machine

			groups. The main problem with their algorithm is that there is a possibility of cell developed without a part family or machine group assigned to it. Another problem that could arise is that more than one part family may be assigned to one cell. Also, the Fuzzy C-mean algorithms work well with smaller data sets and well structured Machine-Part incidence matrices only and clustering errors are observed once the problem size increases and data becomes more and more ill-structured.
3	Zhang & Wang [1992]	FL	In this research FL was applied to the conventional ROC method using input in the form of non-binary Machine-Part incidence matrix. Initially with the help of the conventional ROC method the number of clusters were determined and then using that information a fuzzy based methodology was used to form the part families. The weakness of this approach could be its non-simultaneous nature of grouping parts into families and machines into cells.
4	Leem & Chen [1996]	Fuzzy Clustering Algorithm	Here also the authors developed fuzzy clustering algorithm based on determining the similarity coefficient using input in the form of non-binary Machine-Part incidence matrix while taking alternative process plans into consideration. The algorithm was useful in terms considering alternative processing of parts and minimization of cost involved in intercellular movements. The weakness of the approach could be solving the cell formation problem in a sequential manner (first machine group formation followed by part family formation) rather than simultaneous.
5	Gill & Bector [1997]	Fuzzy linguistics	Here the authors developed an approach for the cell formation problem based on fuzzy linguistics. Here first the information related features of different parts was quantified and then families of parts were formed accordingly. The approach was a useful one as far as quantification of the features of parts, in non exact terms, was concerned. Here also the main weakness could be the non-simultaneous approach of Machine-Part grouping.

6	Susano et al. [1999]	Fuzzy C-mean Algorithm	In this research the authors applied the Fuzzy C-mean Algorithm to the cell formation problem with the intention to overcome the basic short comings of the algorithm (lack of ability to solve large size and ill-structured problems). The main aim of their work was to reduce the infeasibility of the conventional Fuzzy C-mean Algorithm. They did overcome that but with a reduction in GE this indicated deficiencies in this approach.
7	Josien & Liao [2000]	Fuzzy C-mean algorithm	In this research the authors tried to improve the Fuzz C-mean Algorithm initially proposed by Chu & Hayya [1991] but had little success. Here the Fuzzy C-mean Algorithm was integrated with the Fuzzy K-nearest neighbour algorithm while using some commonly used performance measures such as GE, grouping index, number of bottleneck machines and exceptional parts etc.

Recently Li et al. [2007] developed an improved fuzzy clustering method for the cell formation problem overcoming most of the shortcomings being identified in the earlier Fuzzy C-mean clustering algorithms. The authors compared their work with other previous studies in which Fuzzy C-mean Algorithms were proposed i.e. original Fuzzy C-mean Algorithm, Chu and Hayya [1991], Susano et al [1999] and reported that they have outperformed each one of them. It is worth noting that with this approach the authors have been able to bring down the percentage of infeasible solutions developed in case of comparatively large and ill-structured data sets, which remained considerably high in the earlier studies.

Though application of FL to solve the cell formation problem seems very attractive, however, issues like; handling large size ill-structured problems and lack of ability to find optimum solutions for such problems, still need to be tackled.

iii. Simulated Annealing (SA) & Genetic Algorithm (GA):

SA and GA are very effective search techniques that actually replicate natural phenomena. They have shown effectiveness in solving a number of combinatorial optimization problems. Since Machine-Part grouping problem is NP-hard (Yasuda et al. [2005]), therefore, these techniques have been very frequently used in literature to solve this problem.

The first researcher to use SA was Kirkpatrick et al. [1983]. His methodology was actually based on the research being carried out by Metropolis et al. [1953]. The SA algorithm was initially developed to handle the optimization of difficult combinatorial problems with the help of randomization carried out in a controlled manner. SA, in fact, is the replication of the actual annealing process in which a system, at a higher energy level, is allowed to be cooled gradually in a controlled environment till it attains its lowest energy level.

There is a great deal of similarity between the actual annealing process and the iterative algorithms. An iterative algorithm basically consists of solution representation, an objective function, a generation mechanism, and a proper schedule to enforce annealing. In SA the evolution of the whole algorithm depends upon the generation of the initial solution. First an initial solution is generated and then its neighbouring solution is evolved from the initially generated solution. If the neighbouring solution is better than the initial solution then it replaces the initial solution otherwise it is accepted with certain value of probability. With each iteration, the value of initially selected temperature is reduced which consequently reduces the probability of accepting worst solutions. The process is kept continued until the stopping criteria (minimum energy level is achieved).

The work of Lundy and Mees [1986] proved that SA based algorithms have greater probability to reach the global optimum or in its neighbourhood, under certain assumptions. SA has several advantages, when compared to other competitive techniques, e.g. it can be easily and quickly implemented. That is the reason that it has been successfully applied to difficult problems and reasonable results have been attained. Several such examples can be found in literature. Kirkpatrick et al. [1983]

applied SA in the area of computer systems design, Bonomi & Lutton [1984] and Aarts & Van Laarhoven [1985] applied it to solve the Travelling Salesman Problem, Wilhelm and Ward [1987] used SA to solve the Quadratic Assignment Problem, whereas Alfa et al. [1991] applied it to the Vehicle Routing Problem.

Being a strong random search algorithm SA proved to be equally efficient when applied to the cell formation problem, as well. Some of the earliest work carried out by different researchers in designing CMS using SA is presented in Table 3.6.

Table 3.6: SA Based Techniques

S/No.	Reference	Technique	Review
1	Boctor [1991]	SA	The author initially formulated the cell formation problem as a linear mathematical model and then utilized SA to solve the model.
2	Venugopal & Narendaran [1992a]	SA	Here, the authors developed an optimization model for the cell formation problem based on SA with the objective of minimizing the machine load variation.
3	Chen & Srivastava [1994]	SA	In this research a quadratic programming model was developed for the formation of machine cells and SA was used to solve the cell formation problem. The approach, though seems a logical one, is not grouping machines into cells and parts into families simultaneously.
4	Chen et al. [1995]	SA	Here, the authors developed a heuristic based approach for the cell formation problem and used SA to find solution for different problems. The objective of their approach was to minimize the total number of intercellular moves while grouping machines into cells and parts into corresponding families.
5	Boctor [1996]	SA	An SA based algorithm was developed during this research with the objective of minimizing the total manufacturing cost.
6	Sofianopoulou [1997]	SA	Here, the CMS design problem was formulated as the linear integer programming model. The technique had an objective to minimize the total number of intercellular moves while applying a constraint on the size of cell and utilising Sa to find solution for the problems. The

			main advantage of the algorithm was that the number of cells, the system was supposed to be divided into, was not known as a *priori*. But the enforcement of the cell size constraint may result an increase in the number of intercellular moves in case of larger problems having ill-structured Machine-Part incidence matrices. Also, the size of the problems solved is comparatively small.
7	Su & Hsu [1998]	SA	In this research the authors developed an optimization model for the cell formation problem with multiple objectives of minimizing the total cost involved in intercell and intracell moves (transportation cost), intracell machine load variation and intercell machine load variation. A parallel SA was proposed to solve the cell formation problem. The approach is a logical one and can be useful in practice but the main challenge would be to implement it to a real life situation which is normally larger and more complex than the computational experience presented in the paper.
8	Caux et al. [2000]	SA	Here the authors developed an optimization model for the cell formation problem while considering alternative process plans and the machine capacity constraint with the objective to minimize the intercell traffic. To solve the cell formation problem they proposed an SA based methodology and combined it with a branch-and-bound technique for the routing selection. The approach presented is a logical one but the computational experience presented in the paper is very limited which may not be enough to justify its effectiveness when applied to large size practical problems both in terms of accuracy and computational time.

Some of the recent research works in which SA is applied to the cell formation problem includes Xambre and Vilarinho [2003], Tavakkoli et al [2005], Safaei et al. [2008]. Xambre and Vilarinho [2003] developed a mathematical programming approach for the cell formation problem and allowed the duplication of bottleneck machines with an objective to minimize intercellular moves subject to the machine capacity constraint and cell size constraint. Due to the combinatorial nature of the problem they used SA to solve it. The authors besides comparing their work, with other available similar studies in literature, applied it to one practical situation and reported satisfactory results both in terms of accuracy and computational time. The point to note in this research is the constraint on the cell size and the duplication of bottleneck machines. Restricting the cell size means that in certain problems the number of cells may increase than the number of naturally available clusters. Also, the duplication of bottleneck machines may reduce the total number of intercellular moves but would increase the initial cost of machines (multiple copies of one machine type), therefore there must be justifiable cost comparison (material handling cost vs. machine duplication cost) should also be considered.

Tavakkoli et al [2005] solved the dynamic cell formation problem using metaheuristics like SA, GA and Tabu search and compared results in the end. Whereas Safaei et al. [2008] developed a mixed integer programming model for the cell formation problem and used hybrid SA to solve it.

"It is an established fact that cell formation problems belong to the class of NP-hard combinational problems" Yasuda et al. [2005]. A number of optimization algorithms can be found in literature that have the ability to find optimal solution, but only for small- and medium-sized problems. "Their deficiencies are exposed once the problem size gets bigger and the Machine-Part incidence matrices become more and more ill-structured" (Tariq et al. [2009]). These are the reason that GA based search techniques can be frequently found in literature. A review of some of the GA based techniques, available in literature, is presented in Table 3.7.

Table 3.7: GA Based Techniques

S/No.	Reference	Technique	Review
1	Venugopal & Narendran [1992b]	GA	The authors applied GA to the cell formation problem with the objective of considering variations in cell load and minimization of the total number of intercellular moves.

2	Gupta et al. [1995]	GA	Here also a GA based methodology is developed to solve the cell formation problem with the objective of minimizing the intercellular and intracellular moves.
3	Gupta et al. [1996]	GA	During this research the authors improved their previous work [Gupta et al. (1995)] and integrated the cell formation problem with the cell layout design problem.
4	Hwang & Sun [1996]	GA based heuristic	Here, the authors developed a two phased GA based methodology for the cell formation problem. In the first phase GA is combined with a heuristic to identify machine groups, whereas in the second phase the corresponding part families are identified. It is evident from the above description that the approach is not a simultaneous one. The results could have been even better had a simultaneous approach been used instead of sequential.
5	Su & Hsu [1996]	GA	They also developed a two phased methodology for solving the cell formation problem. Here also, one could argue that the results could have been better had a simultaneous approach for Machine-Part grouping (cell formation) been used.
6	Joines et al. [1996]	GA	During this research initially the cell formation problem is formulated as a mathematical model based on integer programming and later on GA was used to solve the optimization problem.
7	Alsultan & Fedjki [1997]	GA	Here the authors formed part families by using the combination of quadratic integer programming model with GA and then later on found corresponding machine groups. The sequence of events shows that a simultaneous approach for Machine-Part grouping has not been used and therefore the results may have been even better had somehow the Machine-Part grouping been carried out simultaneously.
8	Lee et al. [1997]	GA	The distinguishing feature of this GA based approach is the consideration of routing flexibility.
9	Gravel et al. [1998]	GA	The authors developed a double-loop GA based approach for the cell formation problem while considering routing flexibility.
10	Cheng et al.	GA	In this research the cell formation

	[1998]		problem was solved with the help of GA in the same fashion as the travelling salesman problem using GE as the performance measure. Since simple GA was used therefore more than 50% of the results were worst than the previously reported results.
11	Moon and Gen [1999]	GA	The authors, initially, developed the cell formation problem as a 0-1 integer programming model and used GA to solve it. They tried to develop independent machine cells by considering routing flexibility and dual copies of bottleneck machines. The approach is a logical one but can be applied where there is reasonable cost justification for the duplication of machines.
12	Moon & Kim [1999]	GA	Here the cell formation problem was initially developed as a 0-1 integer programming model with the objective of maximizing the total number of intracell moves while considering the cell size constraint. The model was later on solved by using GA. The approach is an effective/logical one, but considering the cell size constraint creates other complications e.g. avoiding the formation of single machine cells, in most of the cases, results an increase in total number of intercellular moves which consequently increases material handling cost [Tariq et al. (2007)].
13	Lee-Post [2000]	GA	The GA based approach presented in this paper forms the part families first, considering the similarities encoded in an exiting classification and coding scheme, and then groups machines accordingly. The approach is an effective one but could have proved to be more effective had the two activities (part families formation and machines grouping) been carried out simultaneously.
14	Zhao & Wu [2000]	GA	The authors presented a GA based approach for the machine grouping problem considering multiple objectives such as minimizing cost involved in intracell part movements, cell load variation, and number of intercell movements. The approach is an effective one as the work of some of the previous researchers, have been further improved.

			The obvious shortcomings of the approach are its non-simultaneous (handling machine grouping and part family formation separately) nature and the smaller size of the problems solved.
15	Arzi et al. [2001]	GA	During this research the authors developed a mixed integer programming model for the cell formation problem and employed GA to solve problems of larger size.
16	Dimopoulos & Mort [2001]	Genetic Programming (GP)	Here, the authors developed a Genetic Programming (GP) based approach for the cell formation problem using the concept of hierarchical clustering. The use of GP for the cell formation problem is unique in itself but the point, in this research, that one can argue about is its limited computational experience and application to comparatively simpler (well structured Machine-Part incidence matrices.
17	Onwubolu & Mutingi [2001]	GA	In this research a GA based approach is developed to solve the cell formation problem by taking into account the cell load variation.
18	Uddin & Shanker [2002]	GA	This work consisted on the formation of an approach based on GA with the objective of minimizing the total number of intercellular moves while considering multiple process plans.
19	Wu et al. [2002]	GA	The GA based approach presented in this paper is an integrated one. Here the cell formation problem is integrated with the cell layout problem.

Some of the comparatively recent approaches in which GA has been used, include Fernando and Mauricio [2002] who developed a hybrid approach by combining GA with an LSH with the objective of maximizing grouping efficacy. The approach have presented a detailed computational experience and observed improvement in GE for 57% of the total tested problems (35) from literature. This improvement was due to the fact that the authors have developed a hybrid approach rather than pure GA. In spite of these improvements it is still felt that had there approach been a simultaneous one the results could have been even better. The same point was proved by Tariq et al. [2009].

Yasuda et al. [2005] developed a grouping GA for the multi-objective (minimization of cell load variation and intercellular moves) cell formation problem. The main advantage of their approach was that the number of cells, the system was supposed to be divided into, was also to be determined by the algorithm. The approach would have been more realistic had it taken processing sequence and routing flexibility into consideration as well. A GA-based concurrent design approach for CM was proposed by Wu et al. [2006]. They integrated the problems of cell formation and group layout generation. The approach is quite effective but as far as the computational experience is concerned it has been applied to problems of small and medium size.

Aaron et al. [2006] developed a hybrid approach for CMS design. They integrated the cell formation problem with the machine allocation and part routing problem. Their solution methodology was based on the combination of GA with large scale optimization techniques. Tariq et al [2006], also, developed a hybrid GA by combining GA with an LSH that further improved some of the results presented in Fernando & Mauricio [2002]. The LSH they developed was so effective that in combination with simple multi-cut point crossover and gene to gene mutation it could still improve the results of some of the problems. In their later work (Tariq et al [2007]) an analysis regarding the handling cost saved by allowing the formation of single machine cells is presented. In this research the formation of cells having only one machine was allowed which consequently resulted in saving some material handling cost by reducing the number of intercellular moves. They further improved their previous work (Tariq et al. [2006]), and produced a comprehensive paper (Tariq et al [2009]) that presented a hybrid GA based methodology and further improved the results of most of the problems solved by Fernando & Mauricio [2002]. This research has proved that in the presence of an effective LSH the dependence of GA on its operators (crossover, mutation, and selection) is somewhat relieved.

Genetic algorithm (GA) and simulated annealing (SA) have proved to be prominent algorithms when applied to the cell formation problem in terms of solution quality, size of the problems handled, and convergence speed. Therefore, these search techniques, in comparison to many traditional techniques, have the capabilities to provide basis for the development of more practically useful cell formation algorithms.

According to Holland [1975] GA has the ability to converge on global optimum or nearly so in a large and complicated search space, under given certain conditions on the problem domain. Since GA operates independently from the objective function of the problem, therefore it gives a designer greater flexibility to interchange different objective functions and also make use of the multi-criterion based objective functions. Also, GA can form machine cells and part families simultaneously therefore they are more suitable for handling real life problems which are normally large in size and complicated in nature. Therefore, it has been time and again mentioned in literature that techniques based on SA in general and GA in particular have outperformed most of the conventional cell formation techniques especially when it comes to large sized and complicated nature of problems.

g) Heuristic Based Approaches:

Apart from mathematical programming based approaches, all of the CMS design techniques discussed so far are based on heuristics. All the AI and array based clustering techniques are basically heuristics, but since their solution approach is general in nature, therefore, they are termed as metaheuristics. Apart from these there are also some other heuristics, developed for the cell formation problem, which do not fit in the exact definition of metaheuristics. Branch and Bound (BB) based algorithms are an example of this class. It was originally developed by Kusiak [1990]. BB algorithm is basically an improvement of the Cluster Identification (CI) algorithm. CI algorithms were actually successful clustering techniques which could only work with perfect Machine-Part incidence matrices that could be divided into completely separable clusters or in other words have no bottleneck machines and/or parts. BB algorithms are actually the same as original CI algorithms but with the induced ability to tackle bottleneck elements. A reasonable review of the heuristic based techniques is presented in Table 3.8.

Table 3.8: Heuristic Based Techniques

S/No.	Reference	Technique	Review
1	Wakhodekar & Sahu [1984]	Machine component Cell (MACE) formation approach	It is one of the earliest methods of cell formation and that is why can handle problems of limited size only.
2	Vanneli & Kumar [1986]	The bottleneck cell minimization approach	The paper provided an approach of minimizing bottle neck machines and/or parts by the method of duplication of machines and/or subcontracting of parts. Though it seems appropriate as far as formation of perfect groups of machines and families of parts is concerned but still there has to be more than a reasonable justification based on cost analysis.
3	Askin & Subramaniam [1987]	A cost based heuristic approach	The author cam up with an approach of designing system on the basis of minimizing cost that involves work in process, material handling and fixed machine costs. The main challenge for such an approach is to implement it to real life situations which are normally large in size and more complicated.
4	Wei & Kern [1989]	The machine score similarity based heuristic	The approach presented was based on the calculation of commonality score to assess similarity between two machines. The approach is very simple to implement but suitable for smaller size of problems only.
5	Al-Qatan [1990]	Branch & Bound algorithm	This technique outperformed the traditional ROC technique but still could not handle large size of problems
6	Wei & Gaither [1990]	A multi-objective based heuristic.	A 0-1 integer programming model is used to develop machine groups and corresponding part families while minimizing the overall cost of producing exceptional parts.
7	Frazier et al. [1990]	A multi-objective cell formation	In this research an approach is developed that could handle

		heuristic.	multiple objectives while solving the cell formation problem. The researchers used a random seed heuristic with non-dominated solution theory. Though the heuristic seems effective but it would have been even more effective had it been applied to real life.
8	Harhalakas et al [1990]	An efficient heuristic based approach for cell formation.	An efficient heuristic based approach for cell formation is presented with discussion regarding its industrial applications.
9	Seifoddini[1990]	A probabilistic model approach.	The author presented a probabilistic approach for the cell formation problem with the intention to overcome the different assumptions to handle the deterministic approach.
10	Logendran[1990]	A heuristic based algorithm.	In this research the author developed a heuristic based approach for the cell formation problem, considering the variations in cell load while minimizing the total moves (intercell+ intracell). The approach is effective enough, but the computational experience presented in the paper is very limited.
11	Boe & Cheng [1991]	A close neighbour algorithm.	In this research many shortcomings of the Bond Energy Algorithm and the Rank Order Clustering approach have been overcome. Though the algorithm is effective enough but it may have proved more effective had GE been used instead of Grouping efficiency as the performance measure.
12	Geoffrey et al. [1992]	Intercell reduction heuristic	The authors developed a heuristic for the cell formation problem based on the objective intercell reduction. The uniqueness of the technique is that several performance measures - such as machine utilization, queue length, flow time etc. - have been used. It would have been even more effective had GE been used as

			performance measure too.
13	Kusiak [1992]	Branch & Bound algorithm	Out performed nine other algorithms but the deficiency was still to handle large size problems
14	Cheng [1995]	Branch & Bound algorithm	The author claimed to have produced a more reliable approach that produces optimal results. This speaks itself about the limitation of the method as optimal solutions can only be obtained for smaller size of problems. He compared his results with McCormick et al [1972], Slagel et al. [1974] and ROC[1980]
15	Caux et al. [2000]	Simulated Annealing (SA) based algorithm	In this approach the part routing problem was handled by Branch & Bound approach, whereas the system was designed using SA approach. The question left open was that whether such multi domain solution can be obtained for a real life/large size problem?

3.3 Evaluation of solutions in GT:

Several objectives for the evaluation of solutions in GT have been defined by resaerchers. A list of nine different objectives is given in Ballakur and Steudel [1987]. The most commonly used in literature is the minimization of intercellular material handling costs. It is because of the fact that intercellular material handling cost would be high in case of an inefficient grouping of machines and parts into cells or in other words there would be too many exceptional elements (parts being processed in more than one cell). Other objectives in the list include: maximizing similarities of parts or dissimilarity of machines, minimizing the total amount of production cost, and minmizing the total idle time of machines or maximizing the machine utilization in cells. The objective of the models formulated by Kusiak [1985] and Seifoddini and Wolfe [1987] was to minimize the total number of intercellular moves. On the other hand, the two models presented by Gunasinghe and Lashkari [1989, 1989] had the objective of maximizing the sum of the compatibility index between all machines and parts.

A major defficiency in the objectives, used by many cell formation models, is that they do not have the ability to evaluate the 'goodness' of a solution on absolute basis. Chandrasekharan and Rajagopalan [1989] came up with another measure of performance called grouping efficiency (Equation 3.2).

$$\text{Grouping efficiency} = \eta = q\eta_1 + (1 - q)\,\eta_2 \qquad (3.2)$$

Where:

η_1 = The ratio between the number of 1s in the block diagonal to the total number of elements in the block diagonal.

η_2 = The ratio between the number of 0s outside the block diagonal to the total number of elements outside the block diagonal.

q = Weight factor.

"The major drawback of grouping efficiency is its lower ability to distinguish between good and bad solutions for example, a bad solution with many 1s outside the block diagonal often shows efficiency figures around 75%" (Fernando and Mauricio [2002]). This inability of grouping efficiency increases with increase in the size of the Machine-Part incidence matrix. Therefore, Kumar and Chandrasekharan [1990] proposed another measure of performance called Grouping Efficacy as shown in Equation 3.3.

$$\text{Grouping Efficacy} = \text{GE} = \frac{N_1 - N_1^{out}}{N_1 + N_0^{in}} \qquad (3.3)$$

Where:

N_1 = Total number of 1s in the machine parts incidence matrix

N_0^{in} = Total Number of 0s inside the block diagonal

N_1^{out} = Total number of 1s out side the block diagonal.

The development of GE was actually an attempt to propose a performance measure that does not posses the drawbacks in other objectives especially the grouping efficiency that had the lower ability to discriminate between a good and a bad solution. Contrary to grouping efficiency, the size of the matrix does not affect GE.

The Machine-Part grouping problem is normally solved by block diagonalizing the zero-one Machine-Part incidence matrix, while minimizing intercellular movements and maximizing the utilization of machines inside the cell, simultaneously. "To obtain these two objectives, at a time, GE can be chosen as the measure of performance, because it has the ability to incorporate both the within-cell machine utilization and the intercellular movements, also, it has a higher capability to differentiate between well-structured and ill-structured matrices and finally, it does not require any weight factor as well" (Fernando and Mauricio [2002]).

On the basis of the different advantages described above, GE is used as the measure of performance during this research.

3.4 Summary:

This chapter has given a brief description of the fundamentals of GT and CMS which itself is a conceptual derivative of GT. A comprehensive literature review of the different cell formation algorithms has been presented in section 3.2. The cell formation techniques have been initially divided into two classes as; design oriented and production oriented techniques. Further, the production oriented techniques are distributed into six different types as; graph partitioning, array based clustering, mathematical programming, similarity coefficient, heuristic based approaches, and artificial intelligence. Within each category along with description of the technique a reasonable review of related literature is also presented. Finally a brief overview of some of the renowned performance measures, pointing out their advantages and limitations, is also presented in section 3.3.

All the CMS design approaches discussed during this chapter have certain advantages and disadvantages. Some approaches are very simple as far as their practical application is concerned, for example ROC. Whereas, some have the ability to formulate the CMS design problem more precisely by considering different objectives and constraints, but the problem with such approaches is that they need substantially long computational time to find solutions, for example Mathematical programming. AI-based approaches including ANN, FL, SA and GA have been applied to CMS design, because of their ability to; find solutions in comparatively less computational time, capture and employ design knowledge, handle a number of constraints, utilize several nonlinear performance measures, and simultaneously form machine groups and part families with a lot of ease. Both Heuristic Search and AI-based approaches are relatively new in this area and therefore most of the recent research is utilizing these techniques to handle the cell formation problem.

CHAPTER 4

LITERATURE REVIEW JOB-SHOP SCHEDULING

4.1 Introduction:

Manufacturing industries are the backbone in the economic structure of a nation, as they contribute to both increasing GDP/GNP and providing employment. Productivity, which directly affects the growth of GDP, and benefits from a manufacturing system, can be maximized if the available resources are utilized in an optimized manner. Optimized utilization of resources can only be possible if there is proper scheduling system in place. This makes scheduling a highly important aspect of a manufacturing system. This chapter presents a review of scheduling in general and Job-Shop Scheduling in particular. Finally, a brief review of the scheduling procedures applied to CMS is also given at the end.

4.2 Scheduling:

Scheduling can be defined as, *"the allocation of resources over a period of time to perform a collection of tasks"* (Noor [2007]). Also, another definition of scheduling is that, "*it is a function to determine an actual (optimal or feasible) implementation plan as to the time schedule for all jobs to be executed; that is, when, with what machine, and who does what operation*" (Hitomi [1996]). Scheduling has its applications everywhere, for example; flights scheduling, train scheduling and production scheduling. According to Wiers [1997] manufacturing scheduling is the performance of operations on a set of jobs, with the help of already allocated set of machines, within a specified time.

According to the nature of activities, scheduling can be broadly divided into project scheduling and operations scheduling

4.2.1 Project Scheduling:

It is actually the scheduling of activities involved in carrying out a project. A project can be construction of a factory, a bridge, a high way or maintenance and repair of a factory or a plant etc. A number of software based approaches are available to handle such type of scheduling. Some well known techniques involve; Graphical Evaluation and Review Technique (GERT), Critical Path Method (CPM), Project Evaluation and Review Technique (PERT).

4.2.2 Operations Scheduling:

Operation scheduling can be defined as, "the processing of a set of jobs, in a given amount of time, on the already allocated corresponding set of machines, in a workshop consisting of several machines or production facilities including operative workers" (Hitomi [1996]). Jain [1998] classified the available operations scheduling models as job sequencing, flow-shop scheduling, mixed-shop scheduling, Job-Shop scheduling and open-shop scheduling.

The job sequencing model determines the sequence or order in which a set of jobs would be processed on one machine. For N jobs there are a set of $N!$ number of possible schedules (sequences). From these $N!$ number of sequences, one sequence is selected based on the maximization or minimization of certain objective functions.

"A flow-shop has a typical flow pattern for mass production" (Hitomi [1996]). Here the processing sequence is the same for all jobs. The flow-shop scheduling is carried out by finding out the sequence of machines according to the multiple-stage manufacturing.

In a Job-Shop every job may have a separate processing sequence. "Job-Shop has a typical arrangement for the case of varied production of most jobbing types and batch types" (Hitomi [1996]). The scheduling of Job-Shop is bit more complicated as compared to the flow-shop. Since every job has a separate processing sequence, therefore for each machine a separate job sequence has to be determined and these job sequences should be inter-related with each other in such a way that all the jobs can be processed within the minimum possible time (Makespan minimization).

A mixed-shop is basically the combination of flow-shop and Job-Shop. In this case some jobs have fixed machine sequence like a flow-shop, and some are processed in an arbitrary sequence like a Job-Shop. In other words, "jobs must be processed in a sequence consistent with a given partial order of machines in mixed shop" (Jain [1998]).

The proper sequence of machines is not followed in an open-shop and therefore the processing of jobs can be carried out in any sequence or order. All the models discussed above are actually the derivatives of open-shop model.

In a manufacturing cell, ideally, all the jobs should have similar processing requirements (no intercellular moves), but still the processing sequence may not be the same each jobs. Therefore, a manufacturing cell can be termed as a Job-Shop. Since this research is mainly concerned with the scheduling of manufacturing cells, therefore the main focus will be on Job-Shop scheduling and the rest of the discussion would be only related to this class of scheduling only.

4.3 Job-Shop Scheduling:

"Job-Shop Scheduling Problem (JSSP) is one of the well known hardest combinatorial optimization problems. JSSP being amongst the worst members of the class of NP-hard problems" (Gary and Johnson, [1979]), there is still a lot of room for improvement in the existing techniques. Because of its large solution space JSSP is considered to be comparatively one of the hardest problems to solve. "If there are *n* jobs and *m* machines the number of theoretically possible solutions is equal to $(n!)^m$" (Noor [2007]). Among these solutions an optimal solution, for a certain measure of performance, can be found after checking all the possible alternatives. But the checking of all the possible alternatives can only be possible in small size problems. For example, a very simple problem of 5 jobs and 8 machines will give $4.3\text{x}10^{16}$ numbers of alternatives. Even with a high performance computer, that can evaluate one alternative per micro second, complete enumeration of this problem to find out the optimal solution would take more than 1000 years of continuous processing (Hitomi [1996], Morshed [2006]).

4.4 Solution Techniques to Handle JSSP:

A number of solution techniques to handle the JSSP have been developed over the years. A broad classification of the scheduling techniques is given in Jain [1998]. Initially the techniques are divided into two classes as approximation and optimization techniques. A complete classification is shown in Figure 4.1.

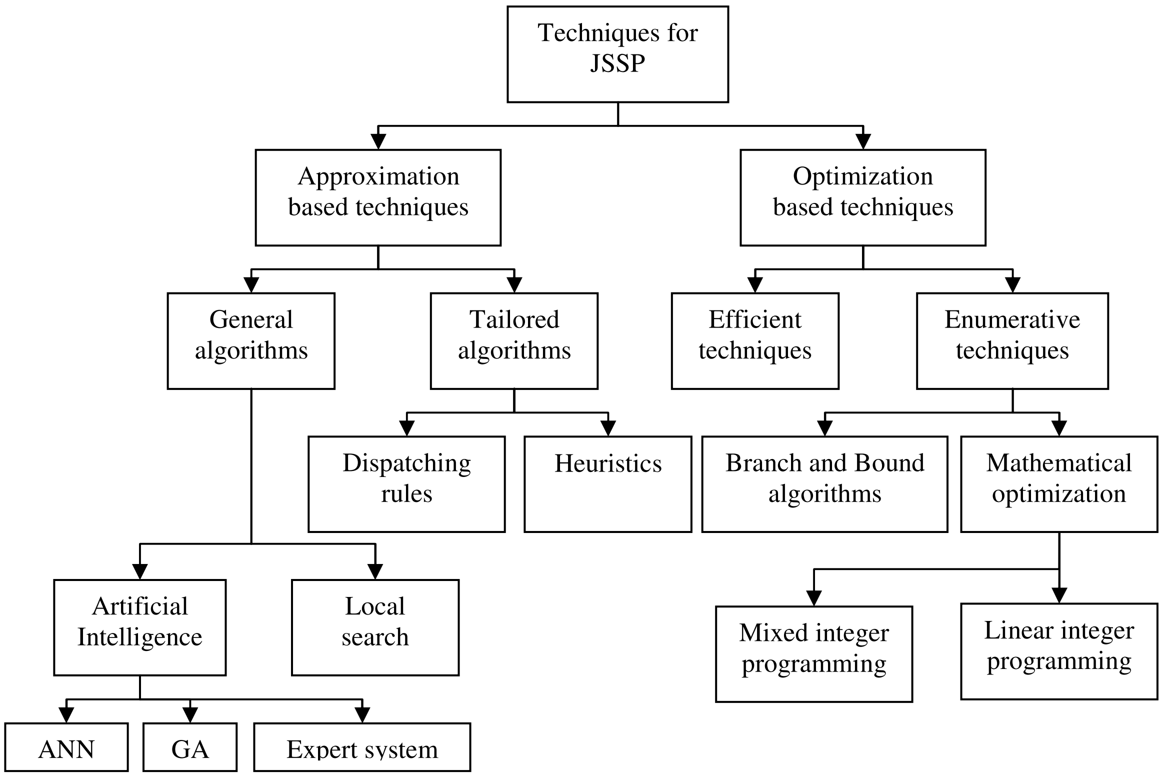


Figure 4.1: Solution approaches to JSSP [Jain(1998)]

Optimization based techniques are further classified as efficient techniques and enumerative techniques. Enumerative approach has further two subclasses as branch and bound algorithms and mathematical optimization (mixed and linear integer programming) based algorithms. On the other side approximation techniques are initially classified as general algorithms and tailored algorithms. Tailored algorithms are either dispatching rules or heuristic based algorithms, whereas general algorithms are classified as AI-based techniques (ANN, GA and Expert Systems) and local search based algorithms. A literature review of the optimization and approximation approaches is given below.

4.4.1 Optimization Based Approaches:

A lot of research work has been carried out, in this area, in the last fifty years. Details are given in the following sections.

4.4.1.1 Efficient Techniques:

Johnson [1954] is one of the earliest research works in this area. He developed a heuristic based efficient method for finding an optimal solution for the two and three stage production scheduling problem, while considering the setup time as well. Akers [1956] developed a heuristic based graphical approach for solving the production scheduling problem. Jackson [1956] extended the Johnson's rule and developed a heuristic based approach for handling the job lot scheduling. Hefetz and Adiri [1982] developed an approach for two machines unit time Job-Shop schedule length problem. These techniques are applicable to very small problems and cannot handle a big problem of more than 3 machines efficiently.

4.4.1.2 Enumerative Techniques:

Mathematical programming based approaches have been extensively used to solve the JSSP. Balas [1965] developed an addictive algorithm for solving the linear programming model that had 0-1 variables. In his [Balas (1969)] further work he developed a mixed integer programming model for machine scheduling using the disjunctive graphs. Mixed integer programming models for the JSSP were also formulated by Mann [1960], Giffler and Thompson [1960] and Balas [1978]. Some of the researchers [Giffler and Thompson (1960), Nemhauser and Wolsey (1988) and Blazewicz et al (1991)] argued that mixed integer programming had not been leading solution approaches to the practical methods of solution. However, the approach presented in Harjunkoski et al. [2000] is a hybrid one, in which they formulated the JSSP as a combination of mixed integer and constraint logic programming.

Another popular enumerative technique is Branch and Bound (BB). It was initially developed by Bellman [1956]. Florian et al [1971], also, developed a BB based algorithm for the machine sequencing problem. The work of Mahon and Florian [1975] was based on BB, too. They developed a methodology to handle the issues of due dates and maximum lateness in JSSP. Whereas, Martin [1991] and Asano and Ohta [2002] proposed heuristics using BB and tried to find optimal solution for the JSSP.

Though mathematical approaches are very attractive as far as formulation of the problem is concerned. But, when it comes to solving the model and finding out the optimal solution of the problem then it becomes extremely difficult as it could require a substantial amount of time. The time requirement increases as the complexity and size of the problem increases. Since JSSP has a higher degree of complexity, therefore mathematical optimization has very limited application as far as large and complex problems are concerned (Noor [2007]).

4.4.2 Approximation Based Approaches:

Approximation based approaches offer a good alternatives for solving the JSSP in terms of the quality of solution and computational time. Though these techniques do not guarantee optimality, but still solutions obtained are feasible and near to optimum, always. Another main advantage of these techniques is the ease with which they can be implemented in practice.

Approximation approaches are further classified as Tailored algorithms (dispatching rules and heuristics) and General algorithms (local search and AI tools), as mentioned in Figure 4.1. A literature review regarding these techniques is presented as follows.

4.4.2.1 Tailored Algorithms:

A problem of N jobs to be scheduled on M machines with the objective of minimizing Makespan (time elapsed between the start of the first operation and the completion of the last operation), C_{max}, can be handled in two steps. In first step jobs are assigned to each machine according to their processing requirements. While, in second step those assigned jobs are sequenced on each machine in such a way that C_{max} is minimized. The second step is normally handled by algorithms known as Tailored Algorithm (Bedworth and Bailey [1987]). Both step one and step two can be followed using a variety of heuristics/sequencing rules for the shop floor schedules for N jobs and one machine and N jobs and M machines (Bedworth and Bailey [1987]). An approach to Job-Shop scheduling or any complex operational scheduling problem can be to break down the main problem into a number of sub problems. Sub problems are scheduled separately with the help of certain algorithms or decision rules. Such methods may not produce guaranteed optimal solution but would definitely present a feasible solution evaluated through a particular performance factor.

A number of decision rules can be found in literature. Some of the renowned ones are described below.

First In First Out (FIFO): In this case job sequencing on a machine is carried out on the basis of the order of their arrival time. The objective behind the first come first serve methodology is to minimize the completion time for individual jobs. This kind of scheduling approach is suitable for service organisations (Fogarty et al. [1991]) and (Vonderembse and White [1991]). A major disadvantage of this approach is that it does not produce consistent results. Momin [1999] observes that the lack of consistent results is because of the fact that the job sequencing is totally dependent upon the probability distribution of their arrival. Another disadvantage is that apart from arrival no other priority is considered. For example, an order released late needs to be moved ahead of other orders in schedule because of certain priority (due date etc.) is not allowed in by this rule (Veilleux and Petro [1988]).

Shortest Processing Time (SPT): According to this rule the priority for job sequencing on a machine is the length of processing time. The shortest the processing time of a job, the earliest the job is to be processed. The objectives associated with this rule are reduction in the: average work in process, average job completion and average job lateness (Smith [1989], Fogarty et al. [1991] and Vonderembse and White [1991]). Though the results of this rule, in terms of its objectives, have been consistent but still one disadvantage is that the job with the longest processing time is always processed in the last.

Earliest Due Date (EDD): As per this rule, jobs are sequenced according to their due dates. The one having the earliest due date is sequenced first and vice versa. The objectives, associated with this rule, are minimization of maximum lateness and average tardiness. Fogarty et al. [1989] argue that this rule works well in the scheduling scenario where most of the jobs have similar processing times. Since priority of job sequencing is their respective due dates, therefore it may be very useful for the companies who are very sensitive about delivery deadlines. However, Vonderembse and White [1991] observed that since one job is processed at a time, therefore it will make other jobs to miss their due dates.

Critical Ratio (CR): Here the job sequencing priority is a ratio, between the time remaining to the work remaining, which is known as critical ratio. Therefore, a job having lowest critical ratio is sequenced first and vice versa. Being a dynamic rule it is mostly used in practice (Veilleux and Petro [1988], Smith [1989], Vonderembse and White [1991] and Vollmann et al. [1997]). The objectives that may be achieved by implementing this rule are minimization of lateness and tardiness.

Management Priority (MP): According to this rule jobs are sequenced according to the priority list provided by the management. The priority may be according to the importance level of a client to the management. According to Momin [1999] the priority is set in advance and provided as an input, related to jobs, in the beginning of a schedule.

The above described dispatching rules have been consistently applied to the scheduling problem in practice as they provide good solutions to complex problems in real-time. One thing that can be concluded from the above discussion is that every rule is suitable for a certain condition and can achieve a certain objective, but when it comes to practice there are a number of other related objectives too, which have to be compromised on. For example in case of SPT rule a job with a longest processing time would always be processed last no matter how urgently required. That is the reason that some researchers use more than one rule in combination. This shows that the selection of a dispatching rule or a combination of dispatching rules actually depends upon the type and amount of information that is taken into account while carrying out the scheduling process. Wu [1987] presented a classification of these rules based on the type and amount of information. Class 1 contains simple priority rules that utilizes a particular piece of information, for example, due dates (such as EDD), processing times (such as SPT), and arrival times (such as FIFO). Class 2 consists of combinations of rules from the previous class. The implementation of a particular rule depends upon the shop floor situation. For example SPT and FIFO can be combined in a way that SPT is used until there are only 5 jobs in queue and once it exceeds this limit the system is switched over to FIFO. This may help in preventing the jobs with longest processing times to be processed last, always. Class 3 contains rules that utilize more than a single piece of information related to jobs and normally referred to as Weight Priority Indexes. Every piece of information is assigned a weight according to its relative importance. First an objective function is defined, for example, $f(x)$ = Processing Time of $\text{Job}_i \times \text{weight}_1$+ (Current Time - Due Date of Job_i) $\times \text{weight}_2$. Then, all the jobs are sequenced according to their respective objective function value.

From the classification described above, it is clear that by combining different rules or utilizing different information about the jobs more effective approaches for scheduling can be developed which can fulfil multiple objectives.

4.4.2.2 General Algorithms/ Artificial Intelligence:

Figure 4.1 shows that general algorithms have been initially classified as AI-based approaches and techniques based on local search. According to Jain [1998] the AI-based approaches, that mainly include GA and ANN, have proved to be more efficient especially in the case of NP-Hard problems, where heuristic based solutions are difficult to find. A literature review of AI tools, applied to the problem of JSSP, is presented below.

4.4.2.2.1 Artificial Neural Networks (ANN):

A reasonably detailed introduction of ANN has already been presented in Chapter 3. Therefore, in this section more emphasis would be given to its application to JSSP.

ANN's configuration can be carried out in terms of activation functions, learning processes, feed forward or feed back, and input type. By changing the number of layers, the number of artificial neurons per layer and the algorithm for changing the weights of the interconnections a number of different configurations with different characteristics can be developed. Each configuration may suit a particular situation or problem. The useful applications of ANN have been explored in the area of manufacturing by Zhang and Huang [1995]. They presented a comprehensive review of the applications of ANN in manufacturing, for example in; scheduling, group technology, computer vision, fault detection etc.

A number of researchers applied ANN to the JSSP. Among these Foo and Takefuji [1988] are considered to be the earliest ones. Some of the other researchers who also applied ANN to the JSSP includes; Zhou et al. [1991], Arizono et al. [1992], Satake et al. [1994] , Jain and Meeran [1998], Hagan [2002], Meeran [2003], Noor et al [2005]. Almost all of them have experienced that ANN is data-hungry tool and precision of its results depends on the number of examples presented for its training.

The output of unknown examples is valid if they lie within 20% range of the training examples.

Some researchers (Montana [1992], Hagan [2002], Meeran [2003], Noor et al [2005]) reported that ANN has the tendency of getting trapped in local optimum because of the trajectory-dependant algorithm used for training. That is the reason that some researchers (Montana [1992], Shazly and Shazly [1999], Yeun et al [1999], Sexton and Gupta [2000], Edward and Taylor [2001], Tsakonas and Dounias [2002]) proposed hybrid approaches and their results showed that hybrid approaches performed better than the traditional ANN.

4.4.2.2.2 Genetic Algorithms (GA):

A reasonably introduction of GA has already been presented in Chapter 2. Here its application to the JSSP would be described and some related review would be presented.

As already described, the four basic steps in the application of GA to a problem are: representation, selection, crossover and mutation. A number of research papers have been produced by different researchers showing different representations and selection procedures with a variety of crossover and mutation schemes.

Representation is considered to be the first step in the implementation of GA to a problem. According to Cheng et al [1996] a total of nine different types of representation have been used while applying GA to the JSSP. Details are as under:

Representation based on:

Operations
Jobs
Preference list
Job-pair relation
Priority rule
Disjunctive graph
Completion time
Machines
Random keys

According to Cheng et al. [1996] the first five types are termed as direct representations, whereas, the last four are termed as indirect representations. In all the direct type of representations a production schedule for a JSSP is directly encoded as a chromosome, whereas, in case of indirect representation a sequence of decisions related to scheduling a system (for example dispatching rules) is encoded as a chromosome. Morshad [2006] carried out a comprehensive review of the different representation schemes used by researchers. According to his survey, direct type representation have been used by Nakano and Yamada [1991], Yamada and Nakano [1992], Fang et al [1993], Gen et al [1994], Norman and Bean [1995], Bierwirth et al [1996], Masaru et al [2000],Wang and Zheng [2001], Zhou et al [2001]; whereas, examples of indirect representation can be found in; Falkenauer and Bouffoix [1991], Tamaki and Nishikawa [1992], Della Corce et al [1995], Donrdoff and Pesch [1995] Kobayashi et al [1995], Donrdoff and Pesch [1995], Ghedjati [1999], Cai et al [2002].

After finalizing representation the next step is selection of chromosomes that may take part in crossover and mutation. A number of established selection procedures have already been mentioned in Chapter 2.

Selection is followed by crossover. Cheng et al. [1999] broadly categorised the different crossover operators, used by researchers while solving JSSP, into two classes as adapted genetic operators and heuristic-featured genetic operators. According to Noor et al [2007] the following (Table 4.1) adapted genetic operators have been used in the last two decades.

Table 4.1: Types of Adapted Genetic Operators (Noor [2007])

S/ No.	Crossover type	Proposed by
1	Partial-Mapped Crossover (PMX)	Goldberg and Lingle [1985]
2	Order crossover (OX)	Devis [1985]
3	Cyclic Crossover (CX)	Oliver et al [1987]
4	Position Based Crossover	Syswerda [1989]
5	Order Based Crossover	Syswerda [1989]
6	Linear Order Crossover (LOX)	Falkenauer and Bouffouix [1991]
7	Partial Schedule Exchange Crossover	Gen et al [1994]
8	Subsequent Exchange Crossover	Kobayashi et al. [1995]

9	Job-Based Order Crossover	Ono et al. [1996]
10	Substring Exchange Crossover	Cheng et al [1997]

Gen and Cheng [1997] observed that in case of JSSP permutation-based representations have been very frequently used by researchers in literature which makes it very easy for some mutation schemes; like insertion, inversion, reciprocal exchange mutation, shift mutation, and displacement; to be implemented.

Some well known heuristic-based genetic operators reported in literature include: Giffler and Thompson [1960], algorithm based crossover of Yamada and Nakano [1991] and the neighbourhood search based mutation of Chen [1991].

A number of researchers applied GA to the JSSP and presented different views about its working. For example, Sakawa and Kubota [2000] observed that GA outperformed SA. Also, according to Onwubolu [2000] GA performs more effectively in reaching the optimum solution of a JSSP. Whereas, some researchers that include; Bierwirth [1995], Dorndorf and Pesch [1995], Morshed [2006]; argue that hybrid GA performs better than standard GA. “A standard GA may not be flexible enough for practical applications and this becomes increasingly apparent when problem is complicated and involves conflict and multi-tasking” (Morshed [2006]). Also, “GAs can rapidly converge on possible solutions; they can sample large spaces randomly and efficiently” (Serdar Uckun et. al. [1993]). However, “they are also subject to such problems as genetic drift and premature convergence” (Serdar Uckun et. al. [1993]). Therefore, some researchers developed hybrid GA procedures (Della Croce et al [1994], Fang et al. [1994], Liaw [2000], Zhou et al. [2001], Noor et al. [2006], Tariq et al. [2007]). “The complementary strengths of GA and local search are such that a hybrid framework of GA and local search can achieve more efficient optimization than GA alone and relaxes the dependence on parameters” (Cheng-Fa & Feng-Cheng [2003]).

4.5 Summary:

Manufacturing scheduling is of paramount importance, as an effective scheduling system ensures optimized utilization of resources. Manufacturing scheduling is broadly categorized as project scheduling and operations scheduling. A project may be a construction of a factory, building or bridge etc. Project scheduling is mainly concerned with the scheduling of activities that are carried out in completing a project. On the other hand operations scheduling takes care of the sequencing and scheduling of operations of a set of jobs on a set of machines in such a way that certain predefined objective is either minimized or maximized. The operation scheduling models can be further classified as job sequencing, Job-Shop scheduling, flow-shop scheduling, open-shop scheduling and mixed-shop scheduling.

Job-Shop scheduling has much in common with the scheduling of manufacturing cells in GT and is a well known hardest combinatorial optimization problem. The techniques developed, over the years, to solve the JSSP are initially classified as optimization and approximation based approaches.

The optimization based approaches are either efficient algorithms or enumerative (Branch and Bound algorithms and Mathematical optimization based techniques). A bulk of optimization approaches are based on Mathematical optimization which consist of either linear or mixed integer programming. Mathematical optimization techniques have the ability to incorporate a number of design related information in the model thus formulating the problem accurately. Therefore they are very attractive as far as formulation of the problem is concerned. But because they consume a lot of time in finding out solutions, their use is limited to small size problems only.

The approximation based approaches are broadly classified as tailored algorithms and general algorithms. Tailored algorithms mainly consist of different types of dispatching rules and heuristics, whereas general algorithms include techniques that are based on local search and AI. The application of AI tools (ANN and GA) is considered as a comparatively recent development in this area. ANN and GA have been extensively used in solving the JSSP. Recently, most of the researchers are of the view that hybrid AI tools perform better than traditional AI tools and that is the reason that trend of using hybrid AI tools to solve the JSSP is on the rise.

CHAPTER 5

METHODOLOGY FOR MACHINE-PART GROUPING

5.1 Introduction:

The literature review, in the previous chapters, has pointed out two things in particular. One is that hybrid GA based methodologies have consistently performed better than standard GA. The second is that in case of CMS the focus of the researchers has either been on Machine-Part grouping (cell design) or on cell scheduling. Whereas the combined approaches that take care of both the issues (cell design and cell scheduling) are very rare in literature. This is the reason that a combined approach has been developed during this research which not only carries out the Machine-Part grouping but also takes care of the operational issues (cell scheduling). Separate hybrid methodologies (GA + LSH) are developed both for Machine-Part grouping and cell scheduling. Machine-Part grouping is considered to be the first step in developing the operational design of CMS. This chapter describes in detail the working of the hybrid GA based tool developed for Machine-Part grouping by combining GA with an LSH. The LSH is incorporated inside the traditional GA loop. The best solution in each generation is locally improved by the LSH and the improved solution is placed back into population, so that it can take part in different genetic operations (crossover, mutation, inversion) and produce even better solutions.

5.2 Hybrid GA for Machine-Part grouping:

As mentioned above, the first and most important step in the operational design of a CMS is the grouping of parts into families and machines into corresponding groups so that those parts can be processed alongside in one cell which require similar processing. Therefore, for identifying part families and corresponding machine groups an approach is developed during this research that combines an LSH with GA. LSH takes the best solution from each generation and tries to improve its GE by following a number of well defined steps. The procedure continues until no further improvement is possible. Afterwards, the solution is placed back into the population so that it may take part in different genetic operators and produce even better solutions. Further details of the hybrid approach are given in Section 5.4.

5.3 Cell Formation Problem: (Tariq et al. [2006] & [2009])

The problem of cell formation is solved in a simultaneous manner during this research. "The simultaneous Machine-Part grouping approaches generally produce better results in comparison to sequential approaches, since all decisions are taken at the same time" (Mungawattana [2000]).

During this research the problem of cell formation is solved by block diagonalizing the zero-one initial Machine-Part incidence matrix with the objective of maximizing GE which automatically minimizes the total number of intercellular moves and maximizes the utilization of machines inside the cells. "GE has a higher capability to differentiate between well-structured and ill-structured matrices and it does not require any weight factor" (Fernando and Mauricio [2002]). The basic expression of $\mathrm{GE}=\left\{\frac{N_1 - N_1^{out}}{N_1 + N_0^{in}}\right\}$ was first proposed by Kumar & Chandrasekharan [1990] and was also used by Fernando & Mauricio [2002] as measure of performance.

The initial zero-one Machine-Part incidence matrix is represented by *MP*[*i,j*] and is of order *Machs*×*Parts* (where: *Machs* = Total machines in the system & *Parts* = Total parts in the system). Every entry (a_{ij}) in the Machine-Part incidence matrix can be either "1" or "0".

$$a_{ij} = \begin{cases} 1 \text{ (if a particular part '}i\text{' has one of its operations on machine } j\text{)} \\ 0 \text{ (vice versa)} \end{cases} \qquad (5.1)$$

Mathematical Model (Tariq et al. [2006] & [2009])

For the above description of the cell formation problem a mathematical model can be developed which in fact is the first step towards finding solution for the cell formation problem.

$$\text{Objective function} = \text{Maximize GE} = \text{Maximize} \left\{\frac{N_1 - N_1^{out}}{N_1 + N_0^{in}}\right\} \qquad (5.2)$$

Subject to: $\sum_{k=1}^{NC} x_k \geq 1$ (5.3)

k = 1……. NC (Total number of cells)

x = Total number of 1s in cell k.

Where:

GE = Grouping Efficacy,

N_1 = Total number of 1s in the Machine-Part incidence matrix

$$N_1 = \sum_{j=1}^{M} \sum_{i=1}^{P} a_{ij} \quad \text{if} \quad a_{ij} = 1 \tag{5.4}$$

i = 1……...P (Total number of parts)

j = 1……. M (Total number of machines)

N_0^{in} = Total Number of 0s inside the block diagonal

$$N_0^{in} = \sum_{k=1}^{NC} \sum_{j=1}^{NM} \sum_{i=1}^{NP} b_{ijk} \quad \text{If} \quad a_{ijk} = 0 \qquad \text{then} \quad b_{ijk} = 1 \quad \text{else} \quad b_{ijk} = 0 \tag{5.5}$$

b= Any variable.

NM = Number of machines in cell *k*.

NP = Number of parts in cell *k*.

N_1^{in} = Total number of 1s inside the block diagonal.

$$N_1^{in} = \sum_{k=1}^{NC} \sum_{j=1}^{NM} \sum_{i=1}^{NP} a_{ijk} \quad \text{If} \quad a_{ijk} = 1 \tag{5.6}$$

N_1^{out} = Total number of 1s outside the block diagonal.

$$N_1^{out} = N_1 - N_1^{in} \tag{5.7}$$

The objective function (5.2) maximizes GE which in turn minimizes the total number of intercellular moves by reducing the number of 1s outside the block diagonal (N_1^{out}) and minimizes the total number of 0s inside the block diagonal (N_0^{in}) that results in increasing the within-cell machine utilization. Whereas, constraint (5.3) ensures that at least one part and one machine is allocated to each cell. One other thing that needs to be mentioned here is that information about the number of cells has to be provided in advance so that the system can be divided into that many number of cells. The model ensures that a given set of machines and parts is arranged into a CMS in such a way that the number of bottleneck machines (machines that are required by more than one cell) is minimized and their utilization inside their respective cells is maximized.

5.4 Methodology for the Hybrid GA for Machine-Part Grouping: (Tariq et al. [2006] & [2009])

To find an optimal solution for the cell formation problem described and formulated in Section 5.4, a strategy based on complete enumeration (considering all the possible options) can be developed as shown in Figure 5.1. For each set of machine groups all the possible part arrangements are considered and on termination the optimal solution is found.

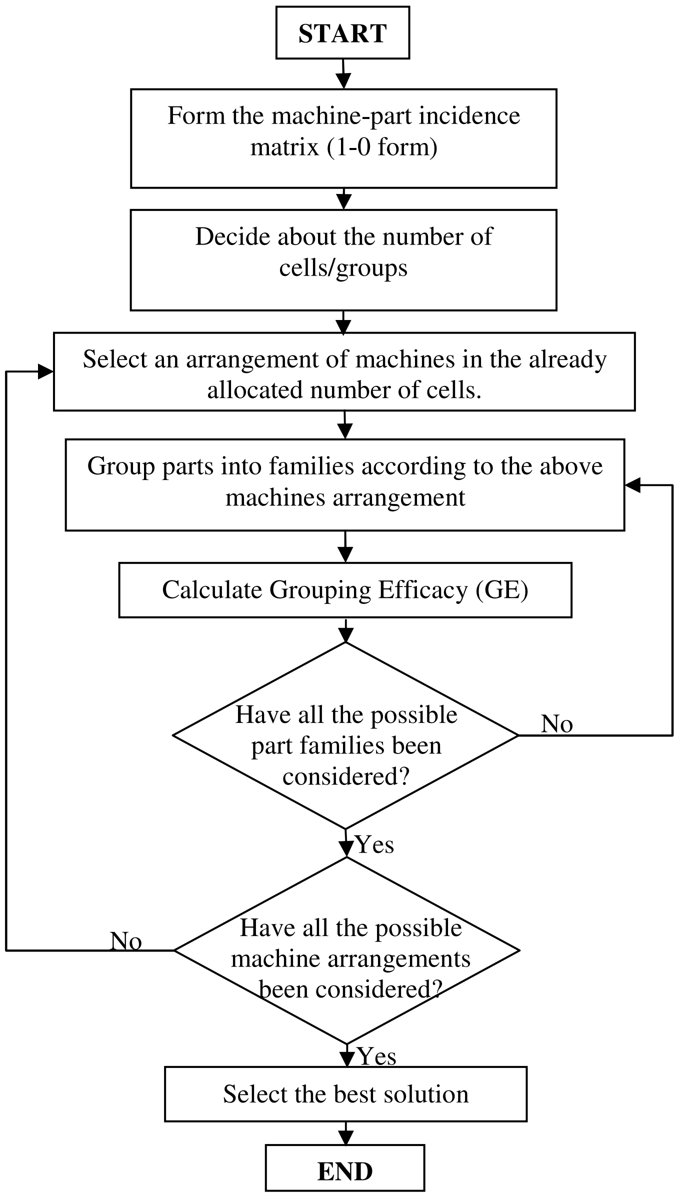

START
Form the machine-part incidence matrix (1-0 form)
Decide about the number of cells/groups
Select an arrangement of machines in the already allocated number of cells.
Group parts into families according to the above machines arrangement
Calculate Grouping Efficacy (GE)
Have all the possible part families been considered?
No
Yes
Have all the possible machine arrangements been considered?
No
Yes
Select the best solution
END

Figure 5.1: Block diagram representation of the proposed Methodology (Tariq et al. [2006] & [2009])

A further stepwise description of the methodology, expressed in Figure 5.1, can be as follows:

i. Arrange the Machine-Part incidence matrix in such a manner that each row represents a machine and each column represents a part. The matrix must contain entries in the form of either 1 or 0. 1 would represent that part *i* has an operation on machine *j* and 0 would mean otherwise.
ii. Decide how many cells have to be developed.
iii. Distribute the total number of machines into the already indicated number of groups (cells).
iv. Tryout, one by one, all the possible combinations of part families and calculate respective GE for each arrangement.
v. Now, repeat step 3, for another combination of machines, and then step 4.
vi. The procedure in steps 3 and 4 must be repeated until every possible arrangement has been considered.
vii. The arrangement that gives maximum GE is to be finally selected.
viii. End.

"The cell formation problem is a combinatorial optimization problem that is NP-hard" (Fernando & Mauricio [2002]). Therefore, complete enumeration can only be possible in case of small size problems. But when it comes to handling problems of large size, it is almost impossible to consider every possible arrangement/division of machines and parts due to rapid increase in computational time and effort. In such a case some kind of search method has to be employed. To cope with this problem a hybrid GA is proposed by combining the conventional GA with an LSH to search for that particular combination of machine groups and corresponding part families which generates a maximum value of GE. The best solution of each generation of GA is selected and further improved with the help of LSH. The process is repeated for 50 generations (Section 5.7) and on termination the best result is selected.

It has been preferred to use a hybrid GA based approach rather than standard GA because of the fact that, over the years, hybrid approaches have generally performed exceptionally well as compared to standard GAs. This thinking was further strengthened by the work presented by Fernando and Mauricio [2002] in which they developed an approach by combining GA with an LSH and as a result there was a substantial improvement in GE for a number of benchmark problems. The hybrid GA developed during this research, also, performed efficiently and further improved the results, presented in Fernando & Mauricio [2002], of different benchmark problems.

All hybrid GAs developed over the years (available in literature) are different from each other in many respects. The hybrid GA developed during this research possesses the uniqueness of having a strong LSH at the heart of the traditional GA loop. The LSH is termed as strong because it produces better results in comparison to all other techniques in spite of the fact that it has been used in combination with the traditional crossover, mutation and inversion techniques. This proves the fact that an efficient LSH relieves a great deal of pressure on the GA operators for producing accurate/better results as would be the case in traditional GA. The LSH proposed during this research is organized in such a way that for a given solution it can change the position of a part and/or machine from one cell to another and observes its effect on the value of GE. Further differentiation between this research and Fernando & Mauricio [2002] is given in Table 5.1.

The hybrid GA based approach developed during this research is presented in Figure 5.2.

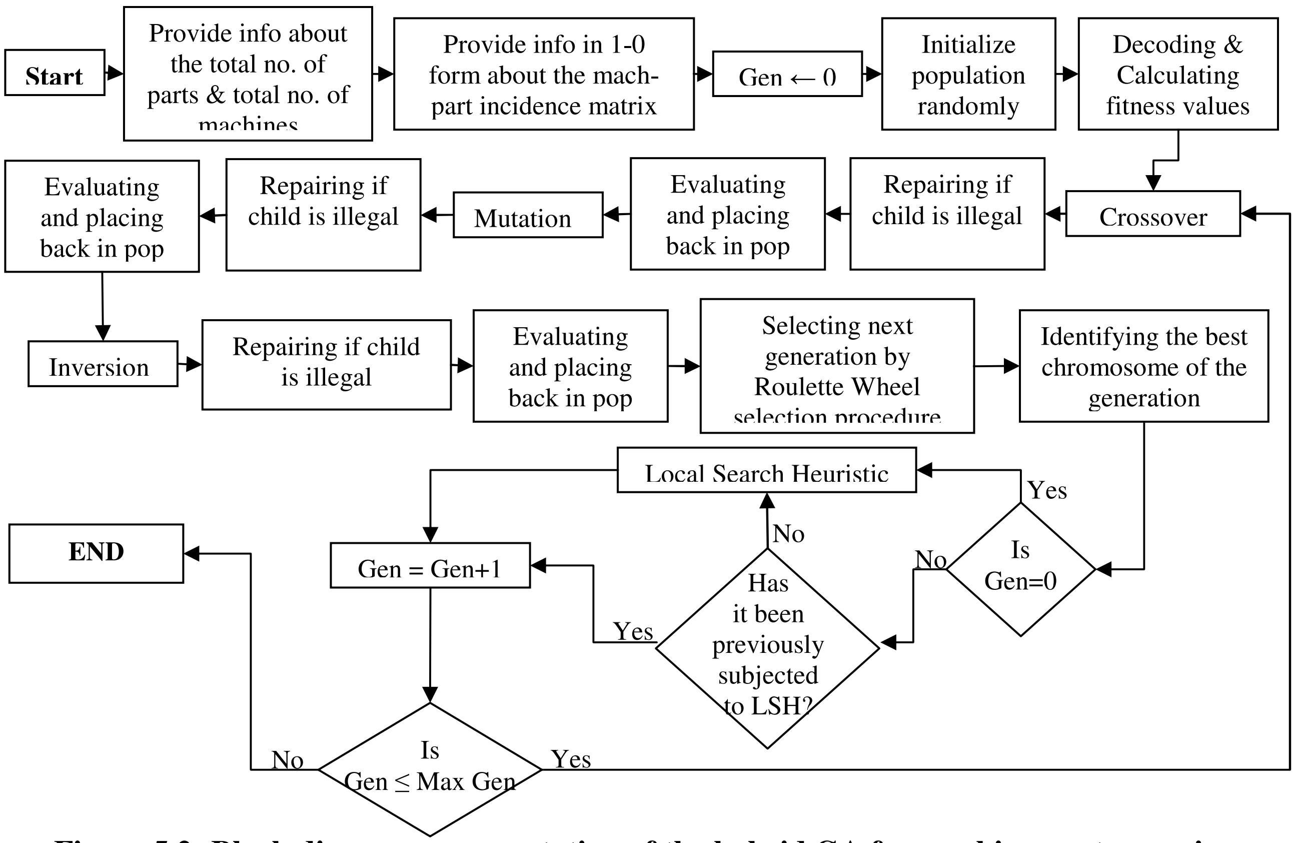

Start
Provide info about the total no. of parts & total no. of machines
Provide info in 1-0 form about the mach-part incidence matrix
Gen ← 0
Initialize population randomly
Decoding & Calculating fitness values
Crossover
Repairing if child is illegal
Evaluating and placing back in pop
Mutation
Repairing if child is illegal
Evaluating and placing back in pop
Inversion
Repairing if child is illegal
Evaluating and placing back in pop
Selecting next generation by Roulette Wheel selection procedure
Identifying the best chromosome of the generation
Is Gen=0
Yes
No
Local Search Heuristic
Has it been previously subjected to LSH?
No
Yes
Gen = Gen+1
Is Gen ≤ Max Gen
No
Yes
END

Figure 5.2: Block diagram representation of the hybrid GA for machine-part grouping

Apparently the approach developed during this research and shown in Fig 5.2 may look similar to the one proposed by Fernando & Mauricio [2002] but actually they are different from each other in many ways. A description of these differences is presented in Table 5.1.

Table 5.1: Differences between the two Hybrid GAs. (Tariq et al. [2009])

S/No.	Fernando's Hyb. GA	Hyb. GA proposed in this research
1	Each chromosome has been represented as a vector of random keys {u (0, 1)}.	Here, each Chromosome is represented by vectors consist of integers. The value of each integer is between 1 and the total number of cells.
2	A chromosome only encodes information about the grouping of machines in each cell. Therefore, its length is equal to *Machs+1,* where *Machs* is the total number of machines in the system. The last gene in a chromosome	A chromosome encodes information both about the grouping of machines into cells and their respective part families. Therefore, its length is equal to *Machs+Parts.* Where *Machs* is the total number of machines and *Parts* is the total number of parts in the system.

	represents the total number of cells the system is going to be divided into.	
3	The type of crossover used is Parameterized uniform crossover. Where at each gene a biased coin (with a probability of tossing heads = 0.7) is tossed to decide that from which parent this gene is going to be selected from.	Here the conventional multi-cut point crossover is used. Four cut points, two in the machines' portion and two in the parts' portion, are randomly selected. This helps in interchanging entries between the same portions of two chromosomes for example entries from machines' portion of one chromosome are interchanged with entries of the machines' portion of another chromosome. This helps in reducing the possibility of illegal solutions produced in the process.
4	Here one or two random solutions are inducted into population in order to maintain a specific level of diversity.	The conventional gene to gene type mutation is used.
5	LSH is proposed for the formation of respective part families for each group of machines provided by GA. The combination is further refined by maximizing the value of GE.	LSH is proposed to be placed inside the conventional GA loop. The best solution of each generation is further locally improved by changing the placement of parts and/or machines with the objective of maximizing the value of GE.

5.4.1 **Genetic Algorithm (GA)** (Tariq et al. [2006] & [2009])

"GAs are stochastic search techniques based on the mechanism of natural selection and natural genetics" (Irani [1999]). The procedure of GA is started with a set of solutions, randomly generated, known as population. Each solution in the population is known as a chromosome. The evolution of chromosomes is carried out through successive iterations termed as generations. In each generation a selected number of chromosomes are subjected to different operations, for example inversion, crossover, mutation, etc, which are known as GA operators. The evaluation of the entire population is then carried out using some fitness measure. Each chromosomes fitness value decides about its selection into subsequent generations. A complete description of GA and its different operators and procedures has already been given in chapter 2, whereas a general GA procedure is shown in the following:

```
Procedure: Genetic Algorithms (Gen and Cheng, 1997)
begin
 t ← 0;
 initialize P(t);
 evaluate P(t);
 while (termination condition not satisfied) do
    recombine P(t) to yield C(t);
    evaluate C(t);
    select P(t + 1) from P(t) and C(t);
    t ← t + 1;
 end
end
```

5.4.1.1 Representation (Tariq et al. [2006] & [2009])

Representation is the first and most important step in the implementation of GA. During this research each chromosome (solution) is encoded in the form of vectors containing integers. Once the representation scheme to be used is decided; the next thing in line is to decide about the length of chromosome. Since here each chromosome would carry information both about the machines and parts, therefore length of chromosome is equal to the accumulative number of machines and parts, so that a gene is allocated for each machine and each part in every solution. Let us consider an example in which there are 4 machines and 4 parts and they have to be arranged into two cells then the following vector can be a solution to the problem:

$Chrom_j = [\underbrace{1\ 2\ 2\ 1}_{\text{Machines}}\ \ \underbrace{2\ 2\ 1\ 1}_{\text{Parts}}]$

Chrom$_j$ can be more clearly represented as shown in Table 5.2.

Table 5.2 Representation of chromosome *Chrom$_j$* (Tariq et al. [2009])

Machines/ Parts →	M_1	M_2	M_3	M_4	P_1	P_2	P_3	P_4
Gene No. →	0	1	2	3	4	5	6	7
Location →	**1**	**2**	**2**	**1**	**2**	**2**	**1**	**1**
	Machines				**Parts**			

Where $Chrom_j$ is an array of integers containing information about the allocation of machines and parts to particular cells (j = 1 to population size). in this example, vector $Chrom_j$ consists of a total number of 8 locations (from 0 to 7); out of these 8 locations the first half of the locations (from 0 to 3) represent machines, whereas the last half of the locations (from 4 to 7) represent parts. Each machine and part is represented by the location of its corresponding gene e.g. the second gene (1) in $Chrom_j$ represents machine1 (M_2) and the sixth gene (5) represents part1 (P_2). Furthermore, each gene's value (allele) shows the allocation (to be placed in which cell) of a particular machine or part e.g. in the solution shown in Table 5.2 the allocation of M_2 is to cell 2 and P_3 is to cell 1.

5.4.1.2 Initialization:

After taking care of representation schemes and length of chromosomes, then comes the stage when a population of chromosomes has to be randomly generated, accordingly. A stepwise procedure can be devised for the initialization process as follows:

i. Enter the total number of parts (*Parts*) to be handled.
ii. Enter the total number of machines (*Machs*) available for the processing of parts.
iii. Provide information about the total number of cells (*NC*) in which the system is to be divided.
iv. Now generate a random number '*K*', so that $1 \leq K \leq NC$. This would ensure that the system is divided into as many number of cells as specified in step iii. For further clarification consider Table 5.2 that represents a two cell problem and therefore none of the genes is having a value greater than 2 or less than 1. This means that a machine/part is either placed in cell 1 (if the value of its corresponding gene is 1) or in cell 2 (if the value of its corresponding gene is 2).
v. $Chrom\ [i,j] \leftarrow K$.
vi. **IF** the number of genes entered in the *i*th row of *Chrom* [*i,j*] are less than an accumulative figure of total number of machines and total number of parts (*Machs* + *Parts*), **THEN** repeat steps 4 and 5, **ELSE** go to next step. This step ensures that each chromosome, in a population, consists of a

total number of genes equal to the sum of *Machs* and *Parts*. Table 5.2 represents a 2 cell problem that consists of 4 machines and 4 parts, therefore the chromosome consists of 8 genes.

vii. Increment the value of row ($i \leftarrow i + 1$). Incrementing i means that after the random generation of ith solution is completed, we move on to the next (i+1) solution.

viii. **IF** the number of rows filled up so far is less than the population size ($i <$ *Pop Size*), **THEN** repeat steps 4, 5 and 6, **ELSE** go to next step. This step ensures that the number of solutions randomly generated must not exceed the population size limit. It is essential because the population size is an important GA parameter and its value is set after carrying out a sensitivity analysis.

ix. Stop.

The above described stepwise procedure can be more clearly expressed in the form of a block diagram representation as shown in Figure 5.3.

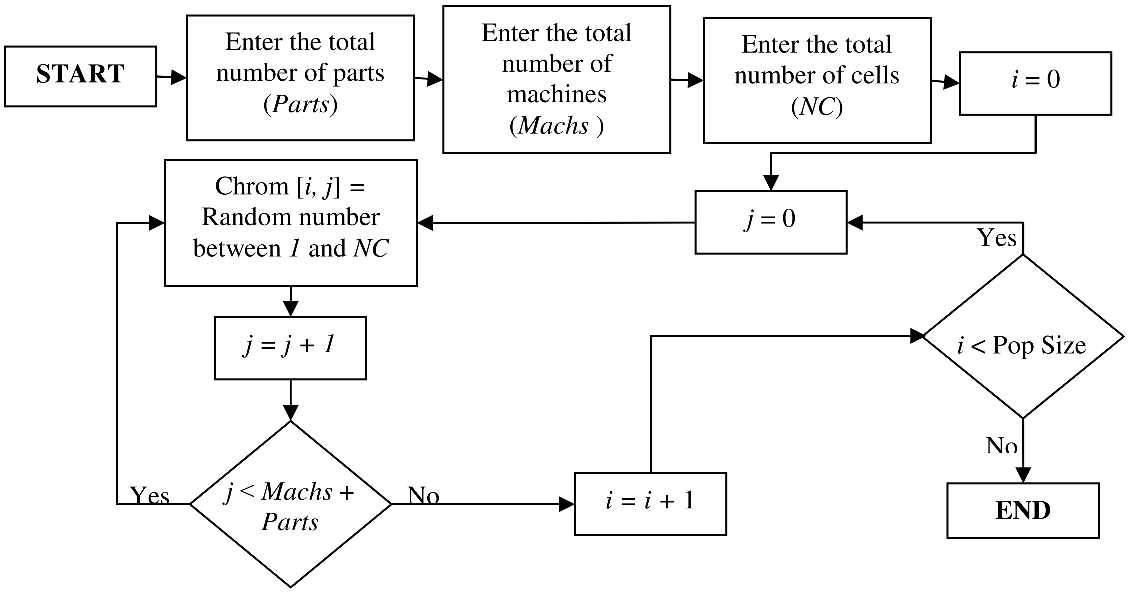

START
Enter the total number of parts (Parts)
Enter the total number of machines (Machs)
Enter the total number of cells (NC)
i = 0
j = 0
Chrom [i, j] = Random number between 1 and NC
j = j + 1
j < Machs + Parts
Yes
No
i = i + 1
i < Pop Size
Yes
No
END

Figure 5.3: Block diagram representation of the initialization process

In Figure 5.3 '*Chrom* [*i,j*]' is a two dimensional array for population, where each row represents a chromosome/ solution. The total number of chromosomes in population (rows in *Chrom* [*i,j*]) depends upon the size of population.

5.4.1.3 Evaluation and Fitness of Solutions in Machine-Part Grouping: (Tariq et al. [2006] & [2009])

It has been described in section 5.4 that on the basis of higher differentiating ability, between the ill-structured and well-structured Machine-Part incidence matrices, GE has been chosen as the performance measure. Since the value of GE has to be maximized, therefore, the fitness function can be similar to the objective function (Equation 5.2). The reason for this is that in case of maximization problems the solution having a higher objective function value is considered to be a comparatively fitter chromosome than the one having lower objective function value. $$ Before selecting the next generation from the current generation, the fitness value of each chromosome is determined. This in turn is responsible for its selection into the next generation. The procedure for evaluating each chromosome consists of the following two steps:

1. First the rows and columns of the initial machine part incidence matrix (MP_{ini} [i,k]) are rearranged according to the arrangement mentioned in a chromosome. For this purpose a blank array, (MP_{fin} [i,j]) of the same size as that of MP_{ini} [i,k], is defined. Then according to the value of each gene a particular row or a column from the initial matrix is copied into the final matrix. Since each chromosome has two portions: machines' portion and parts' portion, therefore first columns of MP_{ini} [i,j] are rearranged and afterwards the rearrangement of rows is carried out. The procedure is started, in the machines' portion, with the minimum value of gene i.e. 1. Starting with the first row (x = 0), in *Chrom* [x,k], the value of each gene is checked. If the k^{th} gene is having value equal to 1 then the k^{th} column from MP_{ini} [i,k] is copied into the j^{th} column of MP_{fin} [i,j], where j is having an initial value of '0' and is incremented by 1 every time a column is copied. The same process is repeated for all the genes in machines' portion having value equal to 1, and after that for all the other values of genes in the same portion, one by one. Once all the genes, having values from 1 to *NC*, in the machines' portion are checked and corresponding rearrangement of columns' carried out, the same procedure is repeated in the parts' portion with the exception that here instead of the rearrangement of columns, rows are selected from MP_{fin} [i,k] according to a gene's value

and copied into another blank array $MP_{fin}1\ [i,j]$. This procedure is clearly represented in Figure 5.4.

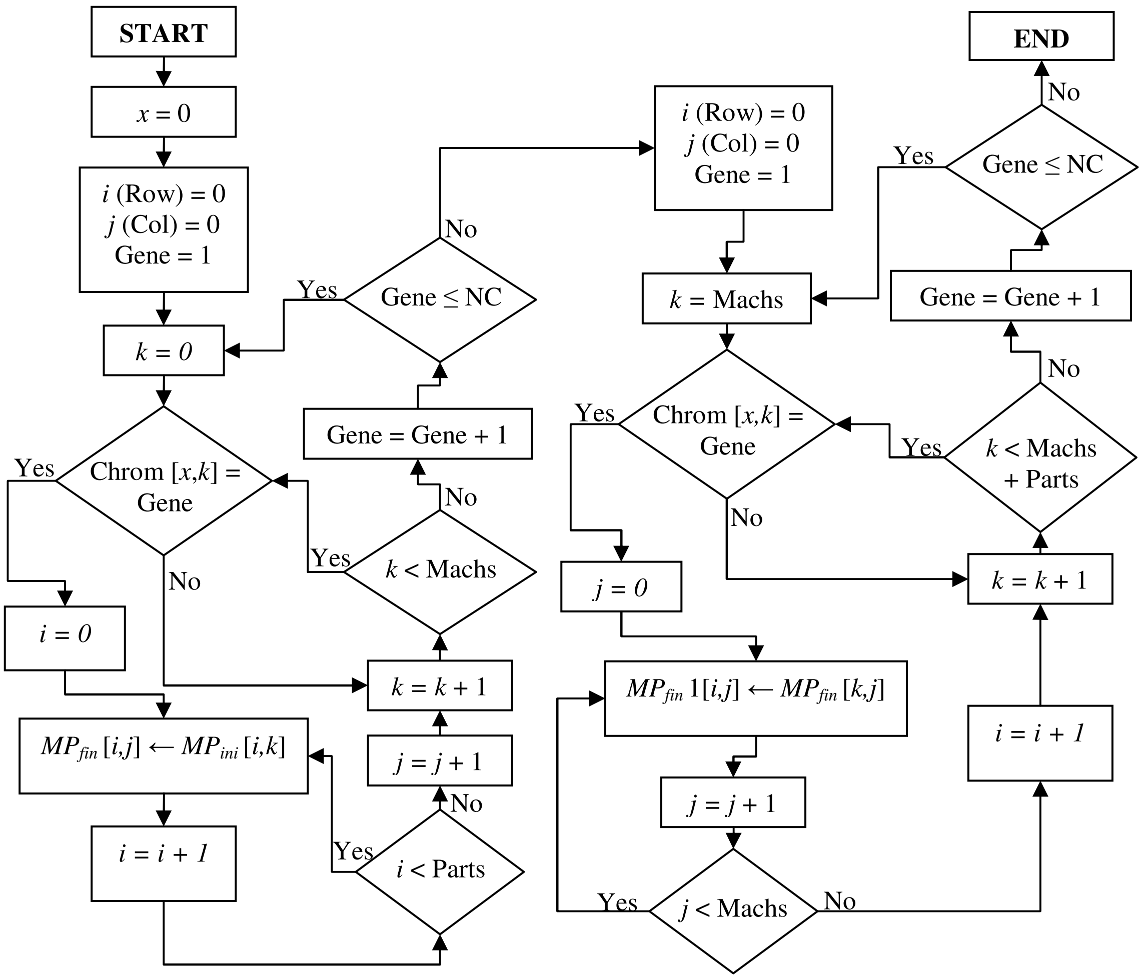


Figure 5.4: The process of rearrangement of machine-part incidence matrix according to a given chromosome

In order to further elaborate the procedure presented in Figure 5.4, a two cell problem having 4 machines and 4 parts is considered as shown below:

Table 5.3: $MP_{ini}[i,k]$

Parts	Machs 1	2	3	4
1	1	0	0	1
2	0	1	1	0
3	0	1	1	0
4	1	0	0	1

Initial matrix

Table 5.4: $MP_{fin}[i,j]$

Parts	Machs			

Blank matrix

Suppose the first row of *Chrom* $[x,k]$ is:

$Chrom\ [0, k] = [\underbrace{1\ 2\ 2\ 1}_{Machs}\ \ \underbrace{1\ 2\ 2\ 1}_{Parts}]$

Now copying columns and rows, respectively, according to the given solution from MP_{ini} $[i,k]$ into MP_{fin} $[i,j]$ and MP_{fin} $[k,j]$ into MP_{fin} $1[i,j]$ as shown in Tables 5.5 and 5.6.

Table 5.5: $MP_{fin}[k,j]$

Parts	Machs			
	1	4	2	3
1	1	1	0	0
2	0	0	1	1
3	0	0	1	1
4	1	1	0	0

Columns are being copied

Table 5.6: $MP_{fin}1[i,j]$

Parts	Machs			
	1	4	2	3
1	1	1	0	0
4	1	1	0	0
2	0	0	1	1
3	0	0	1	1

Rows are being copied

2. Then GE is calculated, with the help of Equation (5.2) after determining values of the variables mentioned in the equation. This step itself consists of several sub steps as follows:
   i. First the total number of operations (N_1) are determined in the given Machine-Part incidence matrix (MP_{ini} $[i,k]$), which is equal to the total number of 1s in the matrix.
   ii. Then from the rearranged matrix (MP_{fin} $1[i,j]$), obtained from Step 1, the total number of machines (NM[k]) and the total number of parts (NP[k]) in cell k is determined.
   iii. Also, from the rearranged matrix the total number of 1s (N_1^{in}) and the total number of voids/ zeros (N_0^{in}), inside the block diagonal, is determined, which in turn helps in calculating the total number of intercellular moves (N_1^{out}).
   iv. Now GE can be calculated by putting values of the variables, found in step (iii), in Equation (5.2).
   v. Steps (ii) to (iv) are repeated population size times in order to calculate GE for all the solutions/chromosomes (Chrom $[x,z]$) in population.

This process, for one solution, is clearly expressed in the form a block diagram representation as shown in Figure 5.9.

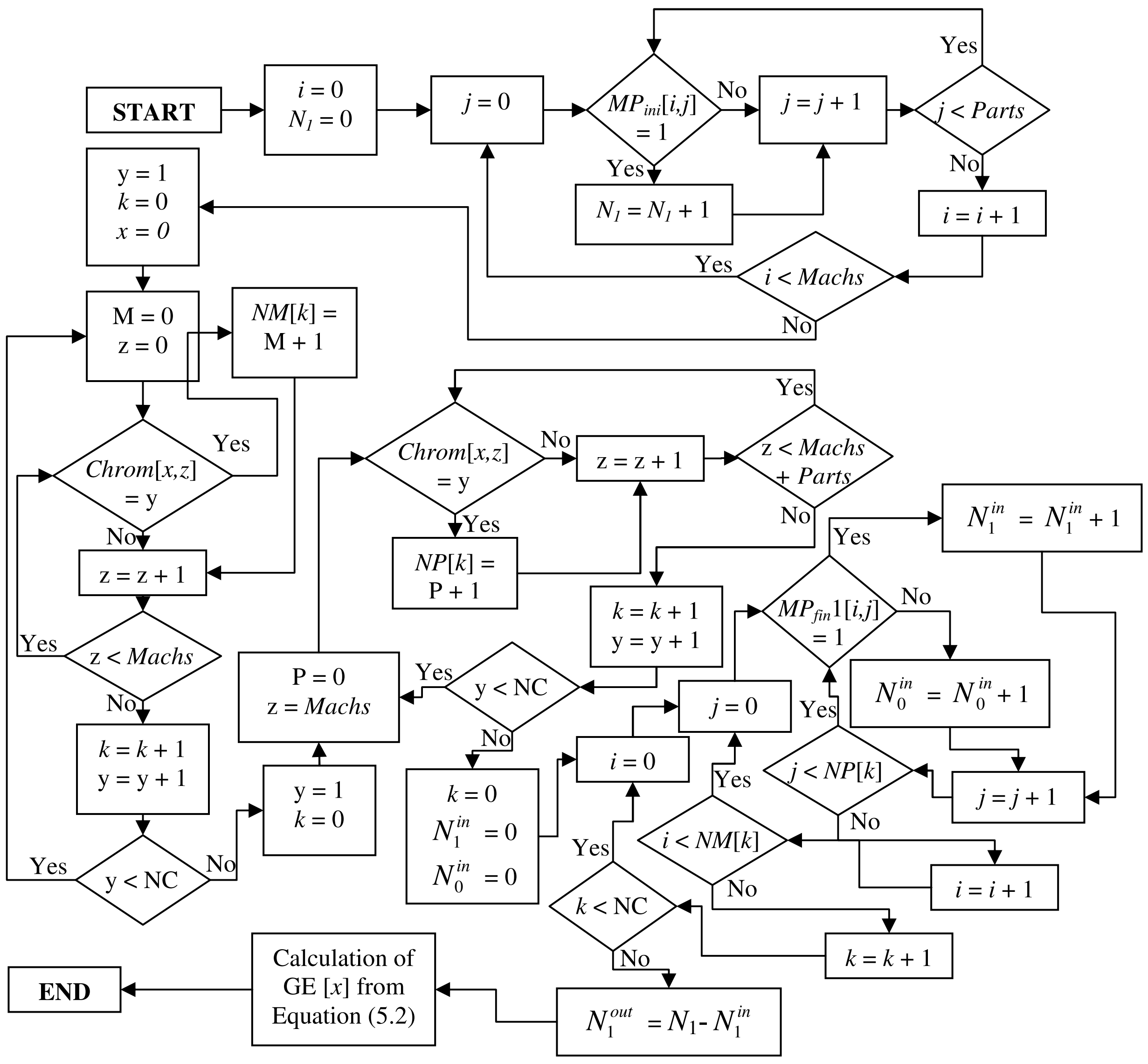


Figure 5.5: Decoding of a Chromosome

Let us consider the same example mentioned in Table 5.3. GE for the problem can be found using Equation (5.2) and the approach presented in Figure 5.5.

From MP$_{ini}$[i,j]:

Total number of 1s in initial Machine-Part incidence matrix = $N_1 = 8$

Now, from $MP_{fin}1[i,j]$:

Total number of machines in cell 1 = 2

Total number of parts in cell 1= 2

Total number of machines in cell 2 = 2

Total number of parts in cell 2 = 2

Using the above information about the number of machines and parts in each cell and the approach presented in Figure 5.5, the following can be found:

Total number of 1s inside the block diagonal = N_1^{in} = 8

Total number of 0s inside the block diagonal = N_0^{in} = 0

Total number of 1s outside the block diagonal = N_1^{out} = N_1 - N_1^{in} = 8 – 8 = 0

Now, putting all the values of variables in Equation (5.2),

$$\text{GE} = \left\{ \frac{N_1 - N_1^{out}}{N_1 + N_0^{in}} \right\} = \frac{8-0}{8-0} = 1 = 100\%$$

5.4.1.4 Genetic Operators

With the help of different genetic operators e.g. mutation, inversion and crossover a chromosome selected initially evolves to many other chromosomes. The proper use of these different GA operators causes improvement in results in each generation and this whole process is stopped when no more improvement is experienced.

5.4.1.4.1 Crossover (Tariq et al. [2006] & [2009])

It is regarded as the major GA operator. Normally, two chromosomes are being subjected to the procedure of crossover, at a time. In this process the features of the two chromosomes are combined together and the generation of two new chromosomes (offspring) is resulted. A crossover approach used during this research is termed as multi-cut point crossover. In this approach initially a set of 4 cut points are selected. The reason for selecting 4 cut points is that every solution consists of two portions: parts' portion and machines' portion. Now, in order to utilize the effectiveness of the multi-cut point crossover and exchange/crossover the elements of the respective parts (machines to machines and parts to parts) a set of two cut points are selected both in the machine and parts' portions of two selected chromosomes. In spite of this arrangement of separate cut points selection in the machines' and parts' portions there is still a possibility of mixing of elements between the two portions. Therefore, a repair algorithm is also developed to tackle such irregularities/ illegalities. Also, 2 cut point crossover can also be used instead of the 4 cut points crossover.

The stepwise procedure used to carryout crossover, during this research, is as follows:

i. Select two chromosomes $Chrom_A$ and $Chrom_B$ on the basis of their better fitness (top two chromosomes)
ii. Generate four random numbers a, b ($0 \leq a < Machs$ and $0 \leq b < Machs$) and c, d ($Machs \leq c < Machs+Parts$ and $Machs \leq d < Machs+Parts$).
iii. **IF** $a < b$, **AND** $c < d$, **THEN** proceed to the next step, **ELSE** go to step 2 and generate random numbers again.
iv. Start from the value of a, i.e. $j \leftarrow a$.
v. Swap the two entries *Chrom* [A, j] and *Chrom* [B, j] in the two selected chromosomes.
vi. Increment the value of j i.e. $j \leftarrow j + 1$.
vii. **IF** $j < b$, **THEN** repeat steps 5 and 6, **ELSE** go to next step.
viii. This time $j \leftarrow c$.
ix. Swap the two entries *Chrom* [A, j] and *Chrom* [B, j] in the two selected chromosomes.
x. Increment the value of j i.e. $j \leftarrow j + 1$.
xi. **IF** $j < d$, **THEN** repeat steps 9 and 10, **ELSE** go to next step.
xii. **IF** the number of chromosomes selected for crossover, so far, is less than 60% of the population size (Section 5.7), **THEN** select the next two best chromosomes in the population and go to step 2 and start all over again, **ELSE** go to next step.
xiii. Stop.

The above described procedure is also shown in the form of a block diagram representation as shown in Figure 5.6.

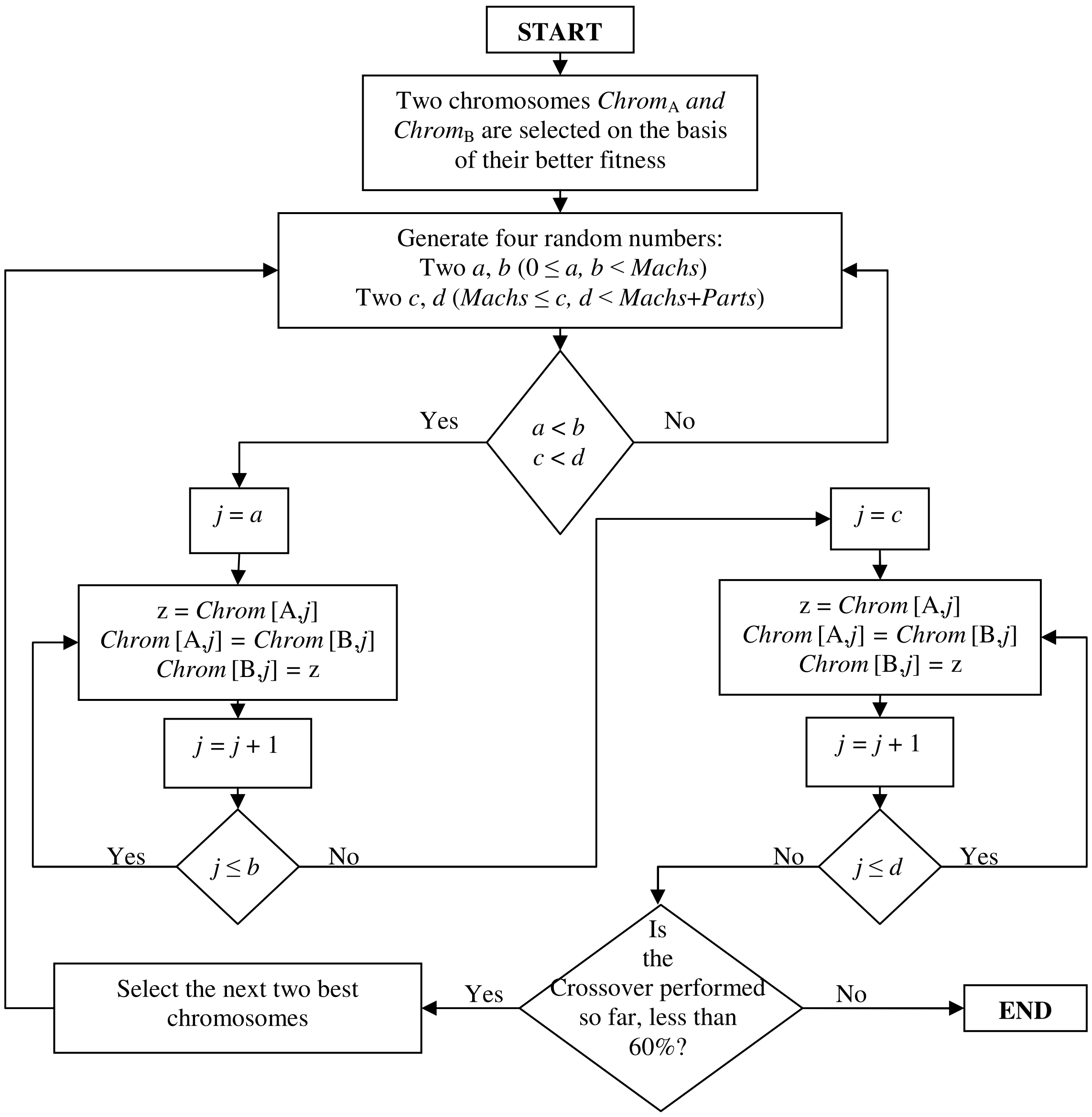

START
Two chromosomes $Chrom_A$ and $Chrom_B$ are selected on the basis of their better fitness
Generate four random numbers:
Two a, b ($0 \leq a, b < Machs$)
Two c, d ($Machs \leq c, d < Machs+Parts$)
$a < b$
$c < d$
Yes
No
$j = a$
$j = c$
z = $Chrom$ [A,j]
$Chrom$ [A,j] = $Chrom$ [B,j]
$Chrom$ [B,j] = z
$j = j + 1$
$j \leq b$
$j \leq d$
Yes
No
Is the Crossover performed so far, less than 60%?
Select the next two best chromosomes
END

Figure 5.6: Block diagram representation of the crossover procedure

In order to further elaborate the procedure presented in Figure 5.6, let us consider the same example of 4 parts and 4 machines. Based on the fitness values the selection of two chromosomes, from the population, is carried out e.g. solution 3 and 5 get selected, as follows:
(Tariq et al. [2006] & [2009])

Chrom [3,*j*] = [1 2 1 2 2 1 2 1]
Chrom [5,*j*] = [**1 1 2 2 2 1 1 2**]

Then a random selection of 4 cut points is carried out (2 each in the machines' and parts' portions of the two selected chromosomes) as follows:

Chrom [3,j] = [1|2 1|2 2|1 2|1]
Chrom [5,*j*] = **[1|1 2|2 2|1 1|2]**

The resulting offsprings, after interchanging entries between the cut points, are as shown below:

Chrom [3,j] = [1 **1 2** 2 2 **1 1** 1]
Chrom [5,*j*] = [**1** 2 1 **2** **2** 1 2 **2**]

The crossover rate used in this research is 60% as discussed in Section 5.7.

5.4.1.4.2 Mutation: (Tariq et al. [2006] & [2009])

It is the GA operator that maintains a certain level of diversity in population by incorporating random and spontaneous changes in selected chromosomes. "In GA, mutation serves the crucial role of either, (a) replacing the genes lost from the population during the selection process so that they can be tried in a new context, or (b) providing the genes that were not present in the population" (Gen & Cheng [1997]). The stepwise procedure adapted, during this research, to carryout mutation is as follows:

i. First, a gene is randomly selected. For this purpose two random numbers i ($0 \leq i < Machs+Parts$) and j ($0 \leq j <$ Population size).
ii. Select the i^{th} gene of the j^{th} chromosome, i.e. *Chrom* [*i,j*].
iii. Now, generate a random number K ($1 \leq K \leq$ NC).
iv. **IF** *Chrom* [*i,j*] = *K,* **THEN** go back to step 3 and generate another random number within the same range, **ELSE** go to next step.
v. Assign the value of K to the initially selected gene (*Chrom* [*i,j*] $\leftarrow K$).
vi. **IF** the number of mutated genes is less than 10% of the total number of genes in population (Section 5.7), **THEN** go back to step 1 and start all over again, **ELSE** go to next step.
vii. Stop.

This procedure is further clarified the form of a block diagram representation as shown in Figure 5.7.

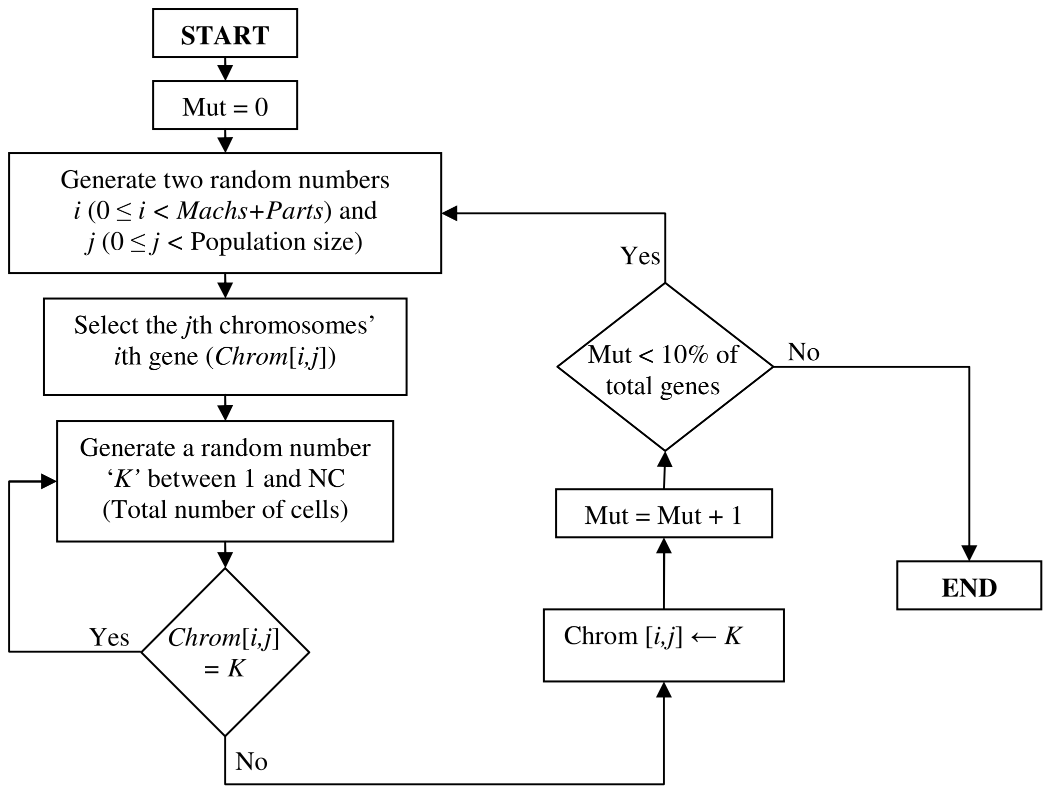

START
Mut = 0
Generate two random numbers
i (0 ≤ i < Machs+Parts) and
j (0 ≤ j < Population size)
Select the jth chromosomes' ith gene (Chrom[i,j])
Generate a random number 'K' between 1 and NC (Total number of cells)
Yes
Chrom[i,j] = K
No
Chrom [i,j] ← K
Mut = Mut + 1
Mut < 10% of total genes
Yes
No
END

Figure 5.7: Block diagram representation of the mutation process

To further clarify the procedure of mutation, being employed here, let us consider an example in which the total number of cells to be developed is 3. Let a gene selected for mutation is *Chrom* [*i,j*] = 1. Now, a random number between 1 and 3 is generated keeping in check that it is not equal to the original value of the gene i.e. 1. Say, the generated value is 2, so *Chrom* [*i,j*] is assigned the value 2 instead of its original value 1. The whole process is repeated until 10% (Section 5.7) genes in a population undergo mutation.

5.4.1.4.3 Inversion: (Tariq et al. [2006] & [2009])

It is the GA operator that inverts a selected portion of a chromosome thus changing its genetic structure. During this research 15% (Section 5.7) chromosomes of the total population have been allowed to undergo inversion. Following is the stepwise procedure being employed to carryout the inversion process:

i. Invert ← 0 (where 'Invert' is a variable).
ii. Randomly select a chromosome '*k*'.

iii. Generate two random numbers i ($0 \le i < Machs+Parts$) and j ($0 \le j < Machs+Parts$).

iv. **IF** $i < j$, **THEN** go to next step, **ELSE** repeat step 3.

v. Swap $Chrom_k$ [i] and $Chrom_k$ [j].

vi. Increment the value of i ($i \leftarrow i+1$) and decrement the value of j ($j \leftarrow j - 1$).

vii. **IF** still $i < j$, **THEN** repeat step 5 and 6, **ELSE** go to next step.

viii. Increment the value of variable 'Invert' (Invert ← Invert + 1).

ix. **IF** Invert < 15% of Pop Size (Section 5.7), **THEN** repeat steps 2 to step8, **ELSE** go to next step.

x. Stop.

The stepwise procedure used is also presented in Figure 5.8, as follows:

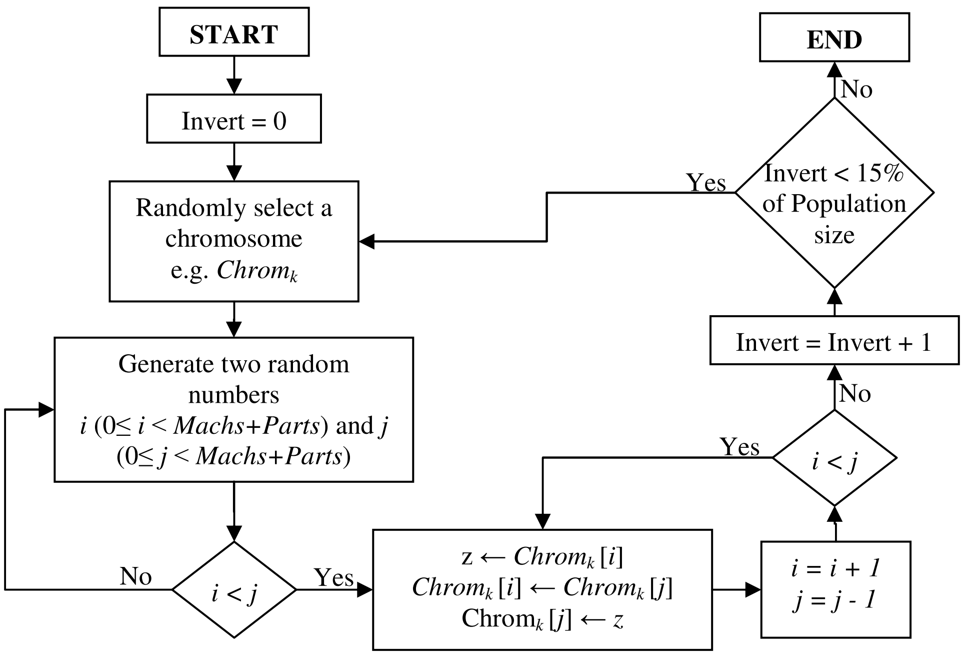

START
Invert = 0
Randomly select a chromosome e.g. $Chrom_k$
Generate two random numbers i ($0 \le i < Machs+Parts$) and j ($0 \le j < Machs+Parts$)
No
$i < j$
Yes
$z \leftarrow Chrom_k[i]$
$Chrom_k[i] \leftarrow Chrom_k[j]$
$Chrom_k[j] \leftarrow z$
$i = i + 1$
$j = j - 1$
$i < j$
Yes
No
Invert = Invert + 1
Invert < 15% of Population size
Yes
No
END

Figure 5.8: Block diagram representation of the inversion process

In order to further elaborate the process of inversion, let us consider an example having 4 parts and 4 machines. Let the following kth chromosome is selected randomly from the population:

$Chrom_k$ [j] = [2 1 1 2 1 2 1 1]

Then two random numbers are generated between 0 and the position number of the last gene which is 7 in this case. Say 2 and 5 are the numbers randomly generated. The inversion procedure is applied in between the selected range as shown below:

$Chrom_k$ [j] = [2 1 **1 2 1 2** 1 1] (the entries within the selected range "positions 2 to 5" are shown in bold)

$Chrom_k$ [j] = [2 1 **2 1 2 1** 1 1] (the entries are shown in inverted form)

5.4.1.4.4 Repair Strategy: (Tariq et al. [2006] & [2009])

The method of encoding (representation) chromosomes and the type of different genetic operators used during this research are such that some illegal/infeasible chromosomes may be resulted. A chromosome is termed as infeasible or illegal if it, after being decoded, generates a solution having one or more cells without a part and/or machine assigned to it i.e. without even a single part and/or machine. The repair strategy ensures that at least one part and one machine is assigned to each cell which is in accordance with the Equation (5.3).

i. Start with $y \leftarrow 1$.
ii. Gene $\leftarrow y$.
iii. **IF** this value of 'Gene' is having at least one entry in both the portions i.e. machines' portion and parts' portion, **THEN** $y \leftarrow y + 1$, **AND** go to step 8, **ELSE** go to the next step.
iv. Now, randomly select another gene ($Chrom_i$ [j]) in the same portion in which 'Gene' does not exist.
v. **IF** $Chrom_i$ [j] exists more than once in the portion, **THEN** go to next step, **ELSE** go step 4 and randomly select another gene.
vi. $Chrom_i$ [j] $\leftarrow$ Gene (the value of a gene that was missing).
vii. Increment the value of y ($y \leftarrow y + 1$).
viii. **IF** $y < NC$ (total number of cells), **THEN** go to step 2, **ELSE** go to next step.
ix. Stop.

The above described repair strategy can be presented in the form a flow diagram as shown in Figure 5.9.

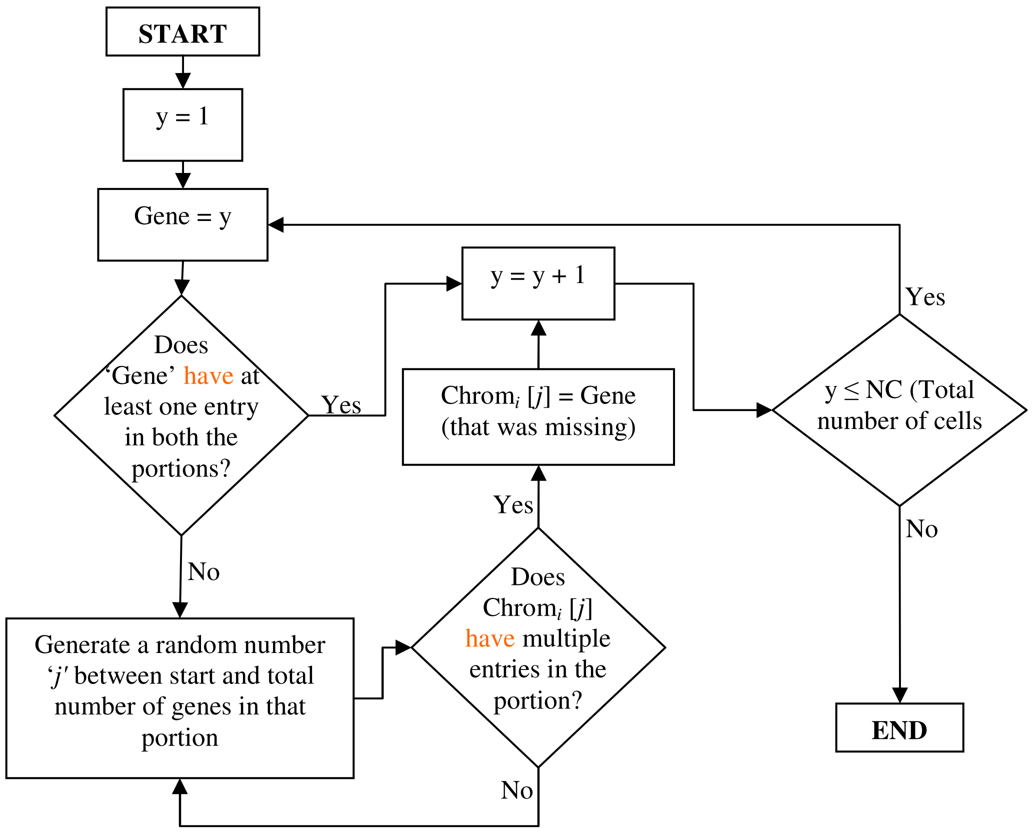

START
y = 1
Gene = y
Does 'Gene' have at least one entry in both the portions?
Yes
No
y = y + 1
Chrom$_i$ [j] = Gene (that was missing)
Yes
Does Chrom$_i$ [j] have multiple entries in the portion?
No
Generate a random number 'j' between start and total number of genes in that portion
y ≤ NC (Total number of cells
Yes
No
END

Figure 5.9: Block diagram representation of the repair strategy

For further clarification an example is considered. It consists of 4parts and 4 machines and they are to be arranged into 3 cells. One of the possible solutions can be as follows:

$$Chrom_i\ [j] = [2\ 1\ 1\ 2\ \ 2\ 2\ 3\ 3]$$

Here, in the machines' portion (first 4 genes in *Chrom$_i$* [j]) integer 3 is not available and also in the parts' portion (last 4 genes in *Chrom$_i$* [j]) integer 1 is missing. So this solution is termed as an illegal one and its repair is required to be carried out. During the repair process a gene, both in the machines' and parts' portion, is selected randomly and then it is checked whether any other gene has the same value as the selected gene. If the answer is "yes" then the selected gene is assigned the value of the missing gene. Let us suppose gene2 and gene6 have been picked. 1 and 3 are the respective values of these selected genes. Since other genes, having the same integer values, do exist in the respective portions of the selected genes there are other genes that have the same values, in their respective portions, therefore, gene2

can be given the value of 3 (since in the machines' portion 3 is missing) and gene6 can be given the value of 1 (since in the parts' portion 1 is missing).
Following is the repaired form of the chromosome.

$$Chrom_i\,[j] = [2\ 1\ 3\ 2\ \ 2\ 2\ 1\ 3]$$

5.4.1.4.5 Selection: (Tariq et al. [2006] & [2009])

During this research the selection approach adopted is based on roulette wheel selection procedure. A roulette wheel selection approach selects a new population using the probability distribution based on the value of fitness. The roulette wheel can be constructed as follows:

i. Start with a counter $x \leftarrow 0$, $GE_T \leftarrow 0$, where GE_T is the total GE of a whole population.
ii. $GE_T = GE_T + GE\,[x]$, where GE $[x]$ is the GE of $Chrom_x$.
iii. Increment the value of x ($x \leftarrow x + 1$).
iv. **IF** $x <$ Pop Size (Population size), **THEN** repeat steps 2 and 3, **ELSE** go to next step.
v. $x \leftarrow 0$.
vi. Calculate selection probability for a chromosome (Sel Prob $[x] = \frac{GE[x]}{GE_T}$).
vii. Increment the value of x ($x \leftarrow x + 1$).
viii. **IF** $x <$ Pop Size, **THEN** repeat step 6 and step 7, **ELSE** go to next step.
ix. $x \leftarrow 0$ and Sum $\leftarrow 0$.
x. Sum $\leftarrow$ Sum + Sel Prob $[x]$.
xi. Now cumulative probability (Cum Prob $[x]$) of a chromosome can be determined by Cum Prob $[x] \leftarrow$ Sum.
xii. Increment the value of x ($x \leftarrow x + 1$).
xiii. **IF** $x <$ Pop Size, **THEN** repeat step 10 to step 12, **ELSE** go to next step.
xiv. $x \leftarrow 0$ and $z \leftarrow 0$.
xv. Generate a random number 'R' ($0 < R < 1$).
xvi. **IF** $R \leq$ Cum Prob $[x]$, **THEN** select $Chrom_x$, **ELSE IF** $R \leq$ Cum Prob $[x+1]$, **THEN** select $Chrom_{x+1}$, **ELSE** $x \leftarrow x + 1$ **AND** repeat step 16.
xvii. Increment the value of z ($z \leftarrow z + 1$).
xviii. **IF** $z <$ Pop Size, **THEN** repeat step 15 to step 17, **ELSE** go to next step.

xix. Stop.

The above described procedure can be clearly represented in the form of a block diagram as shown in Figure 5.10.

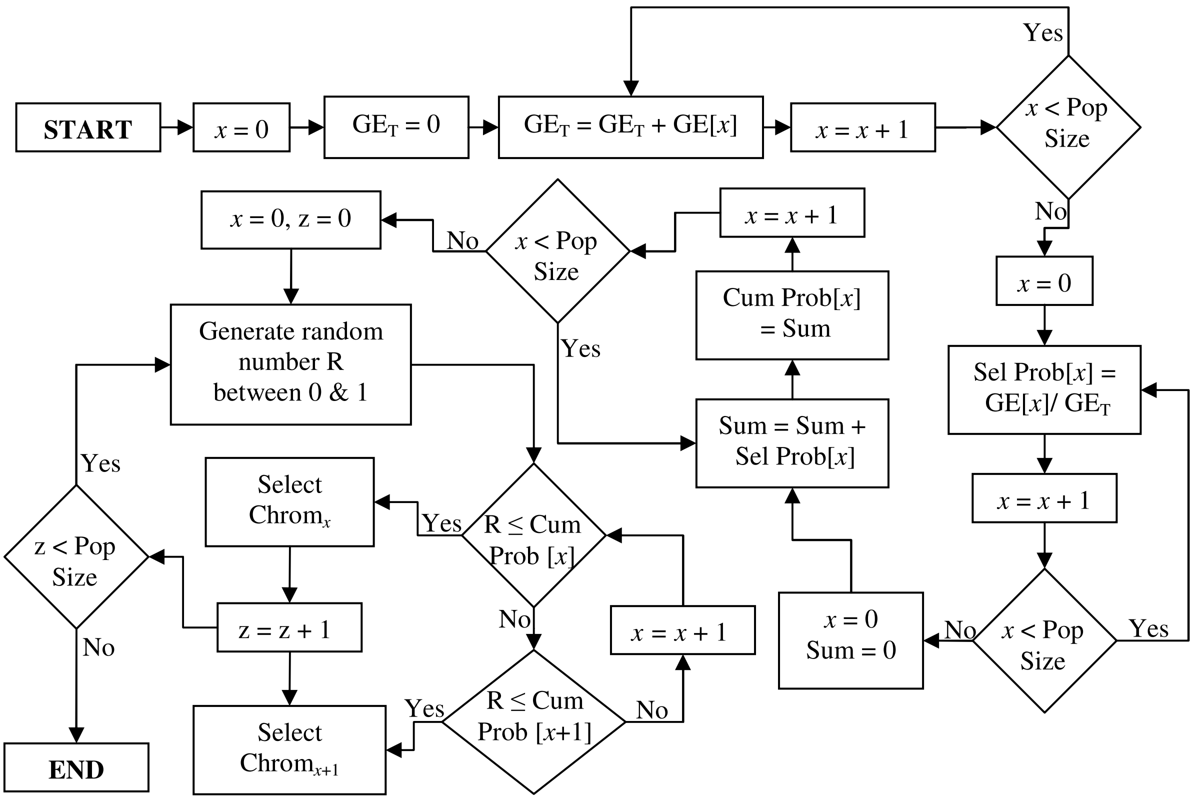


Figure 5.10: Roulette Wheel selection procedure

5.4.2 Local Search Heuristic (LSH): (Tariq et al. [2006] & [2009])

The best solution (Best [*i*]) of each generation is locally further improved with the help of the LSH developed during this research. The actual motivation of developing this LSH is to increase the effectiveness of GA in reaching the optimum or close to the optimum in comparatively earlier generations. The structure of this LSH is such that if by incrementing or decrementing the value of a particular gene the corresponding GE value is increased then such a change is stored and on completion of iteration the same procedure is repeated again. This procedure is kept on repeated until no increase is experienced in a complete iteration. This way the best solution of a generation can be further improved which is most of the times either equal or better than the previously reported best results. Another capability of this LSH is that if a problem consists of a well-structured Machine-Part incidence matrix then it is more likely that the best result would be achieved in the first generation. However, for the problems having ill-structured Machine-Part incidence matrices it may take more than one generation to reach the optimum or close to the optimum. This shows that in spite of the fact that LSH displayed considerable effectiveness; it still depends upon the searching ability of GA to help it out in finding a comparatively better solution (best of the generation) which can be further improved and converted into the best. The development of the LSH procedure in a stepwise manner is as follows:

i. Starting with $y \leftarrow 1$ and $i \leftarrow 0$, the best solution (Best [i]) is checked for any gene having value equal to 'y' in the Machines' portion.

ii. **IF** Best [i] = y, **THEN** $x \leftarrow i$, **ELSE** $i \leftarrow i+1$.

iii. Check for multiple entries of 'y' except at position 'x' (stored value of 'i').

iv. **IF** multiple entries exist **THEN** go to next step, **ELSE** go to step1 and start the same procedure in the parts' portion.

v. Best [x] $\leftarrow$ Best [x] + w, where $w = 1$, **IF** this value of Best [x] is tried previously **THEN** again Best [x] $\leftarrow$ Best [x] + w (this is repeated until a value is found which has not been previously tested), **ELSE** go to next step.

vi. **IF** Best [x] $\leq$ NC (Total number of cells), **THEN** decode Best [i] and calculate GE, **ELSE** go to the start of the procedure, stop incrementing and start decrementing.

vii. **IF** GE > Max GE, **THEN** Max GE $\leftarrow$ GE **AND** Best [x] $\leftarrow$ Best [x] + w, **ELSE** Best [x]$\leftarrow y$.

viii. Repeat steps 5, 6, and 7 until Best [x] + $w \leq$ NC.

ix. **IF** Best [x] + w > NC, **THEN** Best [x] $\leftarrow$ Best [x] – w, **AND** repeat 5, 6, 7 until Best [x] – $w \geq 1$.

x. Keep repeating from 2 to 9, until all the genes having value as 'y' are tested.

xi. **IF** all the genes having value equal to 'y' are tested, **THEN** $y \leftarrow y+1$.

xii. **IF** $y \leq$ NC, **THEN** go to step2, **ELSE** go to next step.

xiii. Has there been any improvement recorded? **IF** yes, **THEN** start all over again, **ELSE** stop.

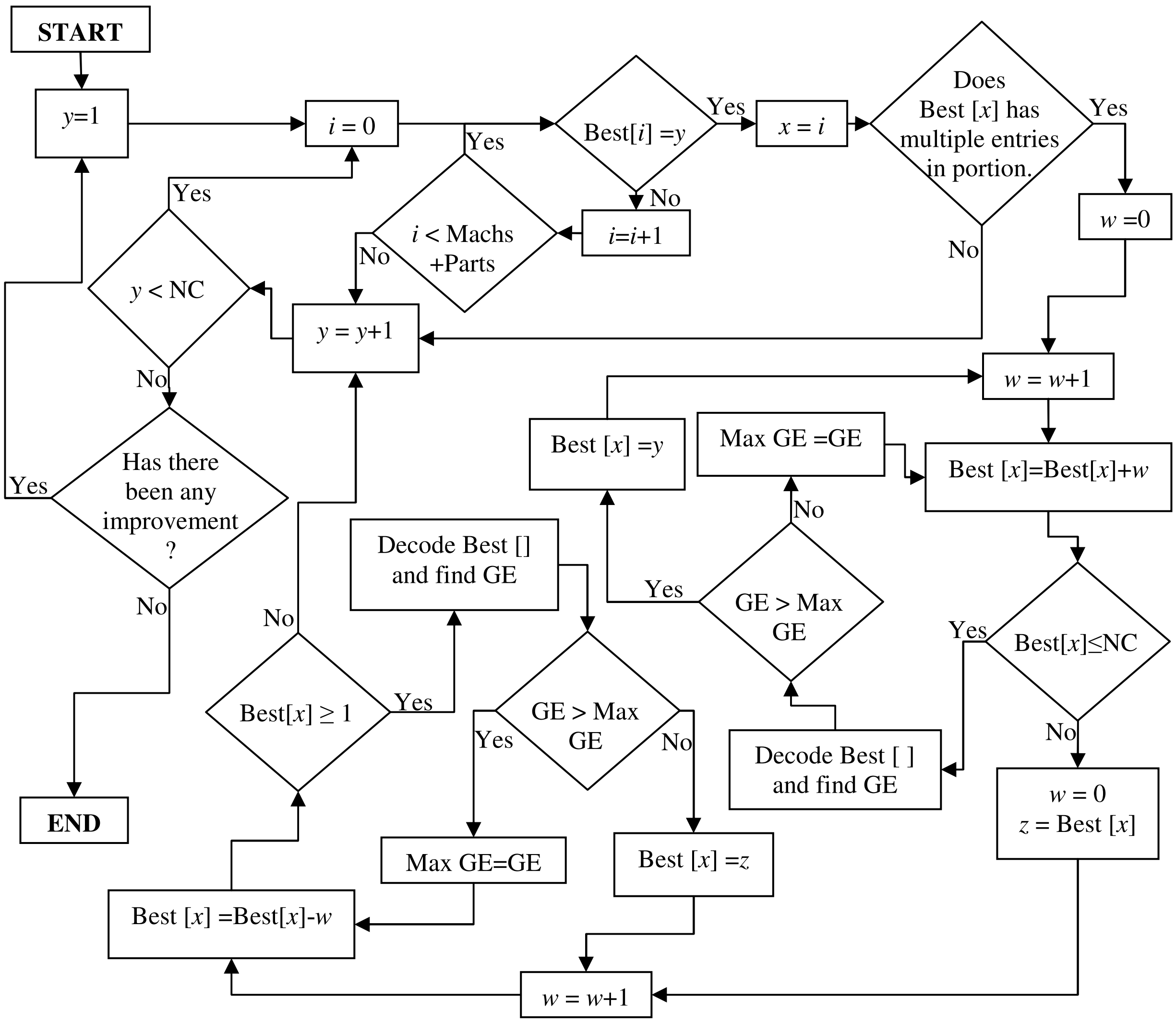


Figure 5.11: Block diagram representation of Local Search Heuristic

Figure 5.11 is the block diagram representation of the LSH, and it further elaborates the stepwise procedure described prior to it.

5.4.3 Numerical Example for Local Search Heuristic (LSH):

To further elaborate the working of the LSH developed during this research an example is selected from Waghodekar & Sahu [1984]. This numerical example consists of 7 parts and 5 machines. Initially the application of LSH in a stepwise manner, to the first gene of the best solution (Best [i]) found by GA in its first generation, is shown in the following. Details of the complete application of LSH to the remaining genes of the solution are presented in Table 5.9. The problem is as shown in Table 5.7.

Table 5.7: Initial Machine-Part incidence matrix (Tariq et al. [2009])

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

This problem is basically about arranging 7pats and 5 machines into a total number of 2 cells while achieving maximum possible value of GE. To achieve this objective the given data is loaded into a computer code developed during this research and based upon the methodology described above. The best chromosome developed by GA in its first generation is as follows.

Best [*i*] = [1 2 2 1 2   1 2 2 2 2 2 1]

The decoded form of the solution Best [*i*] is shown in Table 5.8.

Table 5.8: Decoded solution (Best [*i*]) (Tariq et al. [2009])

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

Using (5.2) (5.4) (5.5) (5.6) and (5.7) we get GE = 62.50%

This solution is subjected to LSH:

Step 1: The procedure is started with $y = 1$, and $i = 0$, where y is any variable and i is the counter for array Best [*i*].

Step 2: All the genes of Best [*i*] are checked for the value of *y*, one by one. Starting with *i* = 0, the value of *i* is incremented by 1 until Best[*i*] = *y*.

Best [*i*] = [1 2 2 1 2   1 2 2 2 2 2 1]

In the above example at *i* = 0 i.e. Best [0] = *y* = 1, as shown in grey colored background. So that value of *i* is stored in another variable *x* and the gene is selected.

Step 3: Now, the machines' portion of the array Best [*i*] is checked for any gene, other than at position *x* (which is the location of the selected gene), that has the same value as *y* (1).

Step 4: Since Best [3] = 1, it means the value of the selected gene can be incremented/decremented in order to change the position of the machine (machine1) related to this gene (gene 0). Had there been no multiple entries of the selected gene, any change in its value would have led to the creation of an illegal chromosome as that would have violated the inequality (5.3).

.**Step 5:** So, Best [*x*] = Best [*x*] + 1 = 1 + 1 = 2, where *x* = 0. This new value of Best [*x*] satisfies the constraint Best [*x*] + 1 ≤ NC, where NC is the total number of cells, which is 2 in this case, so the new chromosome, as a result of the change in the selected gene's value, is shown below:

Best [*i*] = [2 2 2 12   1 2 2 2 2 2 1]

Step 6: On decoding the chromosome, mentioned in step 5 and using (5.2) (5.4) (5.5) (5.6) and (5.7) it is found that GE = 55.56%.

Step 7: Since GE found, as a result of the change, is less than the maximum GE (Max GE) found so far therefore the change is reverted back i.e. Best [*x*] = 1.

Step 8: Now we move to step 5 and again Best [*x*] = Best [*x*] + 1. Since this value of Best [*x*] i.e. 2, is already tested so this time Best [*x*] = Best [*x*] + 2. But since Best [*x*] + 2 > NC, so further incrementing is stopped.

Step 9: Now Best [*x*] = Best [*x*] – 1 = 1 – 1 = 0. But since this value of Best [*x*] violates the constraint Best [*x*] – 1 ≥ 1, so decrementing is stopped, too.

Step 10: The same procedure of incrementing and then decrementing can be applied to all other genes having the value equal to *y* that is 1 in array Best [*i*].

Step 11: Once all the genes having value equal to *y* (1) are subjected to the procedure explained above, then *y* = *y* + 1 = 1 + 1 = 2.

Step 12: Steps 2 to 11 are repeated until *y* ≤ NC. **IF** *y* > NC **THEN** go to next step.

Step 13: **IF** there has been any improvement recorded in the value of GE, **THEN** the procedure is started all over gain, **ELSE** stopped.

Table 5.9: Complete application of LSH to the best solution found by GA (Tariq et al. [2009])

S/ No	Chromos-ome Best [*i*]	Max GE %	Selected gene's changed value (I)	I≤NC	New Chromos-ome	GE %	GE > Max GE	Re-marks
01	[12212 1222221]	62.50	2	Yes	[22212 1222221]	55.56	No	Max GE>GE change is reverted
02	[12212 1222221]	62.50	3	No	-	-	-	Constraint is violated. So change is reverted.
03	[12212 1222221]	62.50	0	No	-	-	-	Constraint is violated. So change is reverted.
04	[12212 1222221]	62.50	2	Yes	[12222 1222221]	68.00	Yes	Max GE<GE change is stored.
05	[12222 1222221]	68.00	3	No	-	-	-	Constraint is violated. So change is reverted.
06	[12222 1222221]	68.00	0	No	-	-	-	Constraint is violated. So change is reverted.
07	[12222 1222221]	68.00	2	Yes	[12222 2222221]	60.71	No	Max GE>GE change is reverted
08	[12222 1222221]	68.00	3	No	-	-	-	Constraint is violated. So change is reverted.
09	[12222 1222221]	68.00	0	No	-	-	-	Constraint is violated. So change is reverted.
10	[12222 1222221]	68.00	2	Yes	[12222 1222222]	55.17	No	Max GE>GE change is reverted
11	[12222 1222221]	68.00	3	No	-	-	-	Constraint is violated. So change is reverted.
12	[12222 1222221]	68.00	0	No	-	-	-	Constraint is violated. So change is reverted.
13	[12222 1222221]	68.00	3	No	-	-	-	Constraint is violated. So change is reverted.
14	[12222 1222221]	68.00	1	Yes	[11222 1222221]	50.00	No	Max GE>GE change is reverted

15	[12222 1222221]	68.00	0	No	-	-	-	Constraint is violated. So change is reverted.
16	[12222 1222221]	68.00	3	No	-	-	-	Constraint is violated. So change is reverted.
17	[12222 1222221]	68.00	1	Yes	[12122 1222221]	50.00	No	Max GE>GE change is reverted
18	[12222 1222221]	68.00	0	No	-	-	-	Constraint is violated. So change is reverted.
19	[12222 1222221]	68.00	3	No	-	-	-	Constraint is violated. So change is reverted.
20	[12222 1222221]	68.00	1	Yes	[12212 1222221]	62.50	No	Max GE>GE change is reverted
21	[12222 1222221]	68.00	0	No	-	-	-	Constraint is violated. So change is reverted.
22	[12222 1222221]	68.00	3	No	-	-	-	Constraint is violated. So change is reverted.
23	[12222 1222221]	68.00	1	Yes	[12221 1222221]	50.00	No	Max GE>GE change is reverted
24	[12222 1222221]	68.00	0	No	-	-	-	Constraint is violated. So change is reverted.
25	[12222 1222221]	68.00	3	No	-	-	-	Constraint is violated. So change is reverted.
26	[12222 1222221]	68.00	1	Yes	[12222 1122221]	56.00	No	Max GE>GE change is reverted
27	[12222 1222221]	68.00	0	No	-	-	-	Constraint is violated. So change is reverted.
28	[12222 1222221]	68.00	3	No	-	-	-	Constraint is violated. So change is reverted.
29	[12222 1222221]	68.00	1	Yes	[12222 1212221]	56.00	No	Max GE>GE change is reverted
30	[12222 1222221]	68.00	0	No	-	-	-	Constraint is violated. So change is reverted.
31	[12222 1222221]	68.00	3	No	-	-	-	Constraint is violated. So change is reverted.
32	[12222 1222221]	68.00	1	Yes	[12222 1221221]	50.00	No	Max GE>GE change is reverted
33	[12222 1222221]	68.00	0	No	-	-	-	Constraint is violated. So change is reverted.
34	[12222 1222221]	68.00	3	No	-	-	-	Constraint is violated. So change is reverted.

35	[12222 1222221]	68.00	1	Yes	[12222 1222121]	62.50	No	Max GE>GE change is reverted
36	[12222 1222221]	68.00	0	No	-	-	-	Constraint is violated. So change is reverted.
37	[12222 1222221]	68.00	3	No	-	-	-	Constraint is violated. So change is reverted.
38	[12222 1222221]	68.00	1	Yes	[12222 1222211]	69.57	Yes	Max GE<GE change is stored.
39	[12222 1222211]	69.57	0	No	-	-	-	Constraint is violated. So change is reverted.

(Genes marked in grey colored background are the selected genes)

The best solution found by the LSH is shown below.

Best [*i*] = [1 2 2 2 2 1 2 2 2 2 1 1] GE = 69.57%

Decoding the solution Best[*i*], modified by LSH, the Machine-Part arrangement is shown in Table 5.10.

Table 5.10: Decoded solution (Best[*i*]) (Tariq et al. [2009])

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

5.5 Numerical Example of Machine-Part Grouping (Tariq et al. [2006] & [2009])

A numerical example is selected from Irani [1999] and solved by this approach to show its effectiveness. This example consists of a total number of 20 parts and 10 machines. The system is required to be organized into 3 cells. The input information in the form of a Machine-Part incidence matrix is given in Table 5.11:

Table 5.11: Initial Machine-Part incidence matrix (Tariq et al. [2006] & [2009])

Parts	**Machines**									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	1			1			1			
2		1				1			1	
3		1				1			1	
4	1			1			1			
5		1				1			1	
6			1		1			1		1
7			1		1			1		1
8		1				1			1	
9	1			1			1			
10			1		1			1		1
11		1				1			1	
12			1		1			1		1
13			1		1			1		1
14		1				1			1	
15	1			1			1			
16	1			1			1			
17		1				1			1	
18			1		1			1		1
19		1				1			1	
20	1			1			1			

For clarity only 1's are shown in the Table.

Number of parts (P) = 20

Number of machines (M) = 10

Length of chromosome (L) = 20 + 10 = 30 (Each chromosome will have 30 genes)

This data is fed into a computer program, encoded in AM (Applications Manager, [2001]) software and based on the methodology explained above. The best possible solution was obtained in the first generation. The result is as under:

$$\text{Best }[i] = [\underbrace{1\ 3\ 2\ 1\ 2\ 3\ 1\ 2\ 3\ 2}_{\text{Machines}}\quad \underbrace{1\ 3\ 3\ 1\ 3\ 2\ 2\ 3\ 1\ 2\ 3\ 2\ 2\ 3\ 1\ 1\ 3\ 2\ 3\ 1}_{\text{Parts}}]$$

Allocating the machines and parts as per the best solution (Best[i]) the Machine-Part incidence matrix, shown in Table 7, shapes up as:

Table 5.12: Final block diagonal matrix (Tariq et al. [2006] & [2009])

Parts	**Machines**									
	1	**4**	**7**	**3**	**5**	**8**	**10**	**2**	**6**	**9**
1	1	1	1							
4	1	1	1							
9	1	1	1							
15	1	1	1							
16	1	1	1							
20	1	1	1							
6				1	1	1	1			
7				1	1	1	1			
10				1	1	1	1			
12				1	1	1	1			
13				1	1	1	1			
18				1	1	1	1			
2								1	1	1
3								1	1	1
5								1	1	1
8								1	1	1
11								1	1	1
14								1	1	1
17								1	1	1
19								1	1	1

Using (5.4) (5.5) (5.7) and (5.2), respectively, the following can be calculated.

The total number of 1's in the Machine-Part incidence matrix (N_1) = 66

Total number of 0's inside the block diagonal (N_0^{in}) = 0

Total number of 1's outside the block diagonal (N_1^{out}) = 0

$$\text{GE} = \frac{N_1 - N_1^{out}}{N_1 + N_0^{in}} = \frac{66-0}{66+0} = 1.00 = 100\%$$

The final block diagonal matrix, shown in Table 5.12, has a GE of 100%.

5.6 Sensitivity Analysis:

The overall performance of GA is considerably affected by its parameters i.e. number of generations, population size, crossover, and mutation rates. Though it seems beneficial to operate with a large population size through a greater number of generations as it provides an opportunity to explore more solution space and facilitates GA to converge on a global optimum, but at the same time it increases the computational effort of GA. Therefore, proper adjustment of these parameters is very important as it fine tunes the performance of GA and enhances its effectiveness.

In the presence of a strong local search, dependence of GA on its parameters is reduced and that is why results for most of the problems are achieved in earlier generations as compared to simple GA or other competitive algorithms in literature. Therefore, a problem of the size 30×50 (Total number of machines = 30 & Total number of parts = 50) is selected from literature (Stanfel [1985]) for analysis, as it required the maximum number of generations in comparison to other tested problems and therefore seems to be a suitable choice. The problem is solved with various number of generations, population sizes, crossover, inversion and mutation rates.

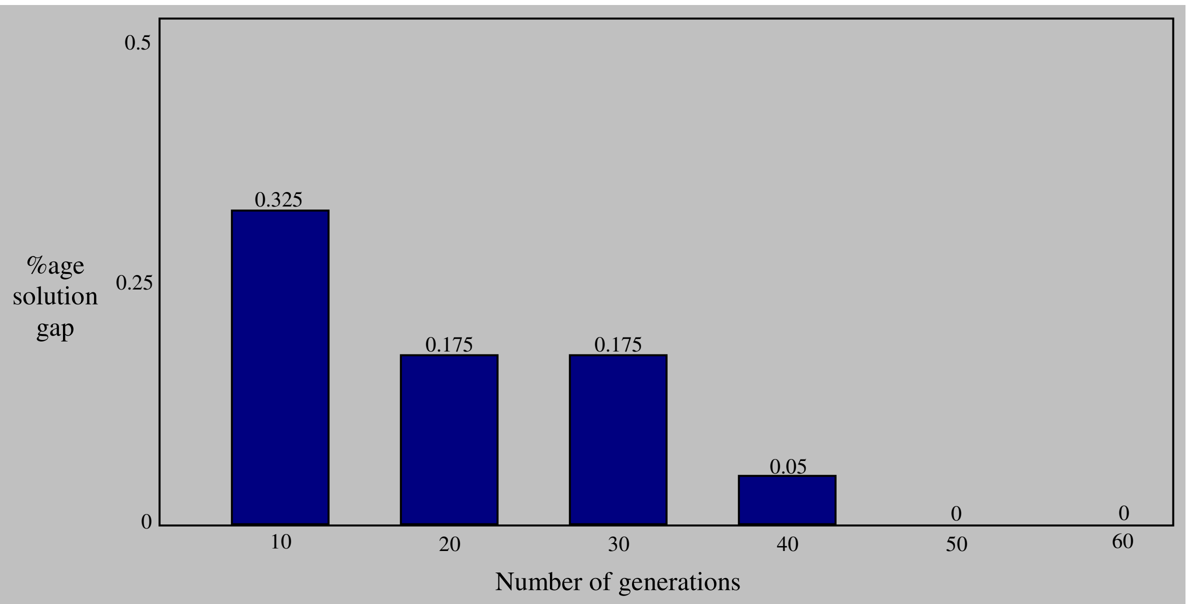


Figure 5.12: Effect of the number of generations on %age solution gap

Figure 5.12 shows the effect of increase in the number of generations on the Percentage (%age) Solution Gap. It can be evidently seen in the Figure that as the number of generations increases the %age Solution Gap decreases and at generation number 50 it reaches to a value of zero. Therefore the maximum number of generations, which the algorithm is allowed to run through, is kept at 50.

The effect of population size on the %age Solution Gap is shown in Figure 5.13, below. Lower bound for the problem is obtained when the population size is increased to 50 with a constant crossover, inversion and mutation rates of 60%, 15% and 10% respectively.

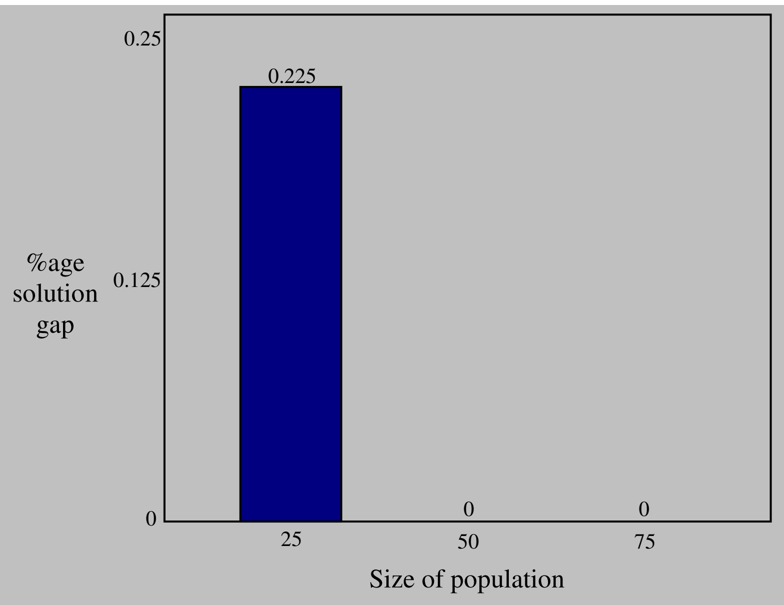


Figure 5.13: Effect of population size on %age solution gap

Number of generations = 50, Crossover rate = 0.6, Inversion rate = 0.15, Mutation rate = 0.1

Further, the sensitivity of the algorithm against variations in the rate of crossover, inversion and mutation is presented in Figures 5.14, 5.15 and 5.16.

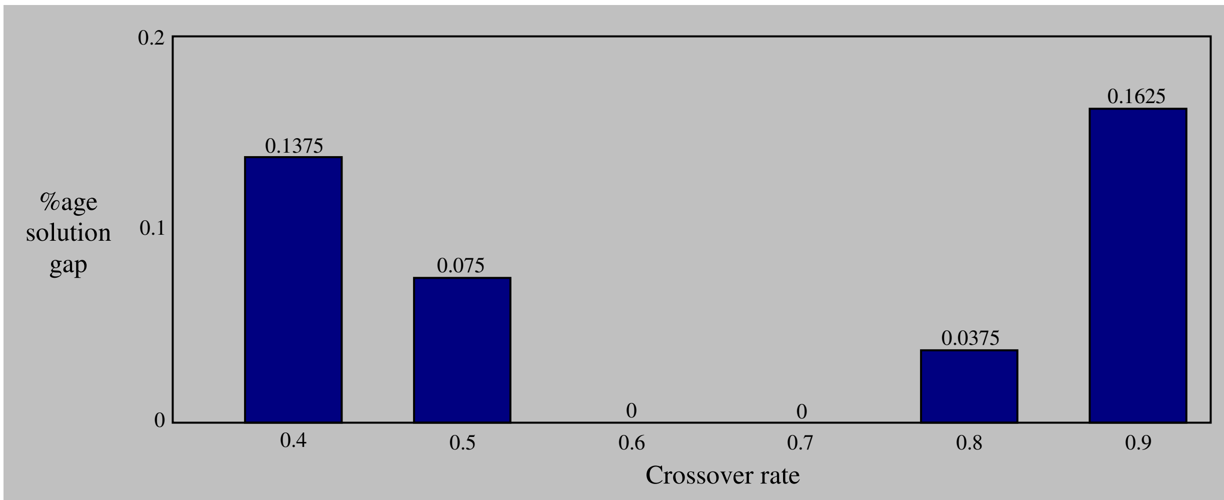


Figure 5.14: Effect of crossover rate on %age solution gap

Number of generations = 50, Population size = 50, Inversion rate = 0.15, Mutation rate = 0.1

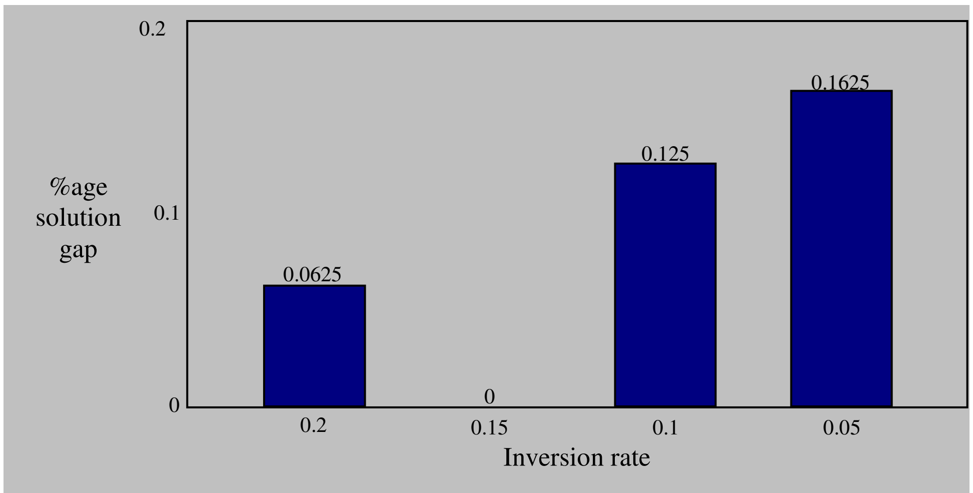


Figure 5.15: Effect of inversion rate on %age solution gap

Number of generations = 50, Population size = 50, Crossover rate = 0.6, Mutation rate = 0.1

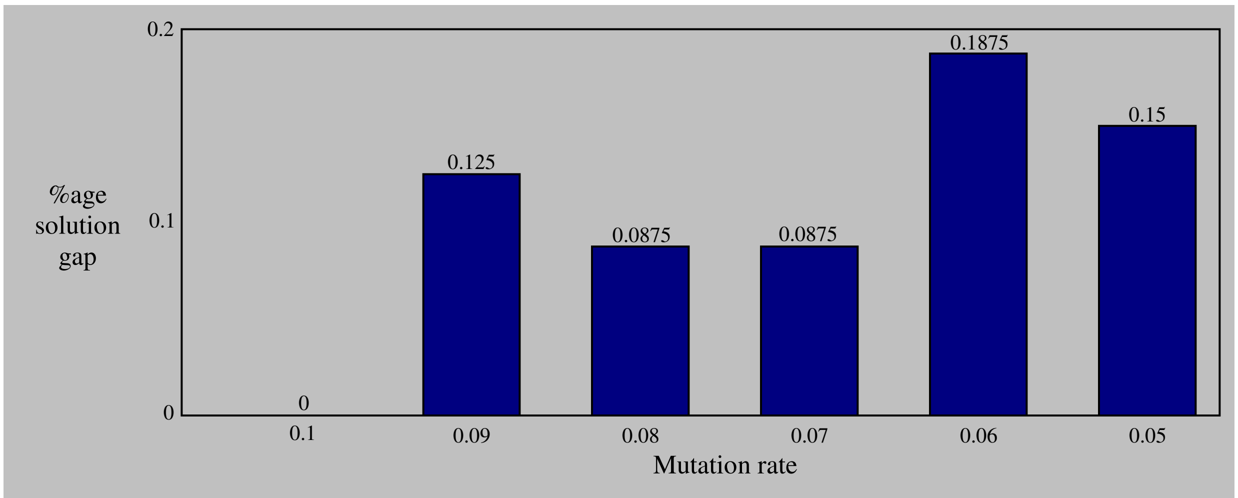


Figure 5.16: Effect of mutation rate on %age solution gap

Number of generations = 50, Population size = 50, Crossover rate = 0.6, Inversion rate = 0.15

From the Figures 5.12, 5.13, 5.14, 5.15 & 5.16 it can be concluded that the algorithm achieved maximum value (zero value of the %age Solution Gap) at a crossover rate = 0.6, inversion rate = 0.15, mutation rate = 0.1, constant population size = 50, and through a constant number of generations = 50. Since the problem used here for analysis consumed the maximum number of generations before reaching its maximum value, therefore it would be more than reasonable to assume that the algorithm with the same set of values for its parameters would perform satisfactorily for other problems as well.

5.7 Summary:

This Chapter has given a detailed description of a hybrid GA based tool that has been developed, during this research, for the Machine-Part grouping problem. It starts with the initial description of the cell formation problem and the mathematical model developed for the problem during this research. Based on the mathematical model an approach is presented, which can be used to find the optimum solution for any problem based on the principles of complete enumeration. Since complete enumeration is possible in problems of limited size only, therefore a GA based hybrid methodology is proposed, during this research, which combines GA with an LSH. The methodology developed utilizes multipoint crossover, traditional gene to gene mutation, inversion, and Roulette Wheel selection procedure. The best solution of each generation is subjected to LSH, which is placed at the heart of the GA loop, provided that it is different from all the previous solutions that underwent local improvement. For better understanding of the programming logic each step of the algorithm is clearly explained both with the help of flow diagrams and stepwise procedure. Finally, the methodology is further elaborated by solving a benchmark problem with the help of the proposed algorithm. In the last part of the Chapter a sensitivity analysis is presented to justify the values of the GA parameters (crossover = 60%, mutation = 10%, inversion = 15%, population size = 50, number of generations = 50).

CHPTER 6

METHODOLOGY FOR CELL SCHEDULING AND COMBINATION WITH MACHINE-PART GROUPING

6.1 Introduction:

It has been previously described that carrying out an operational design of a CMS is comparatively more useful in practice than simply grouping parts into families and machines into corresponding cells. As mentioned in previous chapters that operational design of a CMS consists of two steps (Machine-Part grouping and the cell scheduling). The development and implementation of first step (Machine-Part grouping) has been described in detail in Chapter 5. Description of the second step is given in this chapter that consists of details about the development and operation of the hybrid GA based tool for cell scheduling by combining GA with an LSH and then its combination with the hybrid GA based tool for Machine-Part grouping (Chapter 5). Here also, the LSH is incorporated inside the traditional GA loop in such a way that the best solution in each generation is subjected to it. The effectiveness of the LSH is evident from the fact that though it has been used in combination with the traditional two cut point crossover and swap mutation even then the results produced for all the benchmark problems are as accurate as previously reported in literature.

6.2 Scheduling of a CMS: (Tariq et al. [2007])

As mentioned in section 6.1, the second step in carrying out the operational design of a CMS is to provide solution for scheduling of operations on available machines in each cell. While solving the cell scheduling problem, during this research, it has been assumed that:

- Each job in a cell has its own processing sequence totally independent of the processing sequence of other jobs in the system.
- None of the jobs visits the same machine twice.
- An operation once started cannot be interrupted in between.
- Each machine can process only one job at a time.

It can be easily understood from the above assumptions that the problem of cell scheduling, as far as this research is concerned, resembles the general JSSP and would be handled in a similar manner. Therefore, the methodology developed during this research to solve the JSSP is presented in the following sections, which would be later on applied to the cell scheduling problem after combining it with the solution methodology, already presented, for the cell formation problem.

6.3 A Hybrid GA for JSSP: (Tariq et al. [2007])

The general JSSP is known to be extremely hard and requires efficient, effective and accurate scheduling techniques to realize its full benefits. In this research a hybrid GA is presented to solve *n* jobs and *m* machines JSSP. An LSH is incorporated within GA to optimize Makespan. LSH considerably improves the result of GA and saves a lot of computational effort. The algorithm is tested and verified by using a number of benchmark problems from literature and industrial case studies. For further elaboration of the algorithm solution for a benchmark problem is also presented. The Makespan is used as the main scheduling criterion because of its popular use as performance measure in scheduling and therefore the comparison of most of the different tools/algorithms is made by this measure.

The performance of the approach (GA+LSH) developed during this research is validated through various benchmark problems and industrial case studies. Computational experience with the algorithm proves that in addition to be as accurate as previous approaches it also has the ability to search out the best solution in fewer number of generations.

6.4 The Standard JSSP: (Tariq et al. [2007])

"In the classic *n* × *m minimum- Makespan* JSSP, *n* different jobs on *m* different machines are scheduled" (Gen & Cheng, [1997]). In JSSP every job contains several operations which are to be carried out in pre-specified order. Each operation of the job requires a specific machine on which it is processed for a fixed amount of time (processing time). In addition to the above information a number of constraints, listed below, are also required to be followed.

- There is only one operation of each job on each machine.

- There is no pre-specified sequence of processing between the operations of different jobs.
- Once an operation is started it cannot be stopped in between.
- Only one job is to be processed, at a time, on each machine.
- The delivery time of jobs, i.e. due date or time of release, is not pre-specified.

While following the above mentioned constraints the JSSP is solved by sequencing the operations of all the jobs on the available set of machines with the objective to minimize the total Makespan which in fact is the time elapsed between the start of the first operation and the completion of the last operation.

6.5 Methodology (Developed during this research) for the Hybrid GA for JSSP: (Tariq et al. [2007])

In general it is believed that the best results, as far as the quality of solutions and time are concerned, are achieved from hybrid approximation algorithms, as they combine several methods. From a general perspective, the solution to JSSP can be considered as a collection of local decisions concerning which operation to schedule next. Therefore, the methodology developed in this research is in fact the combination of conventional GA with the LSH (developed during this research) as shown in Figure 6.1. The LSH is incorporated inside the GA loop in such a way that each generation's best chromosome is subjected to it provided that it has not been tried previously. This procedure is kept repeated for a pre-specified number of generations and once it is completed the solution that displayed the best result is selected. This procedure can be described in a stepwise manner as follows:

i. Start with initializing the value of a variable 'Gen' (Gen $\leftarrow$ 0).
ii. Initialize population randomly.
iii. Decode each solution and calculate its fitness value.
iv. Select the best chromosomes for crossover.
v. **IF** children are illegal **THEN** repair, **ELSE** go to next step.
vi. Evaluate children and place them into population.
vii. Randomly select chromosome and carryout mutation.
viii. Evaluate the mutated chromosome and place it into population.
ix. Select the next generation by stochastic universal sampling (SUS).

x. Select the best chromosome of the generation.

xi. **IF** Gen = 0, **THEN** subject the best chromosome to LSH, **ELSE** go to next step.

xii. **IF** the selected chromosome has not been previously subjected to LSH, **THEN** apply the LSH procedure to it, **ELSE** go to next step.

xiii. Increment the value of 'Gen' (Gen ← Gen + 1)

xiv. **IF** Gen < Max Gen (Maximum number of generations i.e. 100 as discussed in Section 6.7), **THEN** repeat step 4 to step 13, **ELSE** go to next step.

xv. Stop.

The above described procedure can be further explained with the help of a block diagram representation as shown in Figure 6.1.

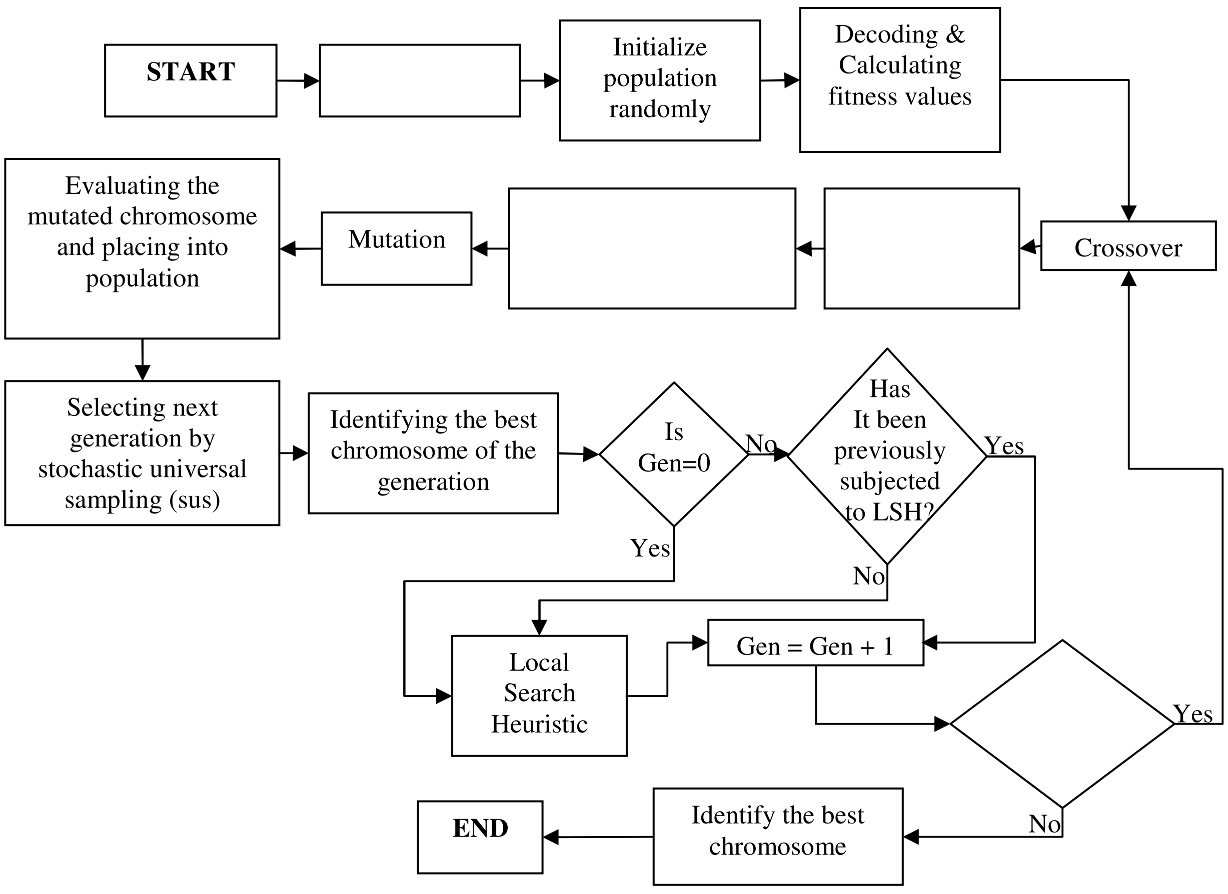

START
Initialize population randomly
Decoding & Calculating fitness values
Crossover
Mutation
Evaluating the mutated chromosome and placing into population
Selecting next generation by stochastic universal sampling (sus)
Identifying the best chromosome of the generation
Is Gen=0
No
Yes
Has It been previously subjected to LSH?
Yes
No
Local Search Heuristic
Gen = Gen + 1
Yes
No
Identify the best chromosome
END

Figure 6.1: Hybrid methodology for JSSP (Tariq et al. [2007])

6.5.1 Genetic Algorithm (GA): (Tariq et al. [2007])

In the hybrid GA, developed during this research, GA starts with an initial population size of 75, crossover rate of 60%, mutation rate of 10% and is allowed to run through 100 generations (details are given in Section 6.7).

6.5.1.1 Representation: (Tariq et al. [2007])

In the hybrid methodology, proposed during this research, chromosomes/solutions are represented in the form of two dimensional arrays of integers as shown in Table 6.1. The size of chromosome is determined by multiplying the total number of machines with the total number of jobs i.e. $Machs \times Parts$. For example, if there are 6 machines and 6 parts and each part has six operations i.e. an operation on each machine then every solution/two dimensional array would be having 36 entries, as shown in Table 6.1.

Table 6.1: Chromosome representation (Tariq et al. [2007])

Direction to read entries in a solution ↓

3	**3**	**1**	**5**	**4**	**3**
4	**2**	**1**	**2**	**5**	**5**
1	**4**	**6**	**6**	**6**	**4**
2	**2**	**5**	**1**	**5**	**1**
6	**6**	**4**	**2**	**2**	**5**
4	**3**	**3**	**1**	**6**	**3**

Every location of the two dimensional array is filled with an integer having value from 1 to 6 i.e. $1 \leq$ integer ≤ 6. Also, the frequency of existence (how many times an integer can exist in a solution) of an integer must be equal to 6. The reason for this is that the maximum number of operations on a job can be 6.

While retrieving information, about the processing schedule of each operation, from a solution entries are taken into consideration along the direction of arrow i.e. row wise, starting from the first row. In the above solution (Table 1) the first entry is '3'. Since integer '3' has been encountered for the first time it means this corresponds to operation 1 of job 3. Moving further, the second entry in the same column is '4'. Since '4' has also been encountered for the first time therefore this also corresponds to the first operation of job '4' and its scheduling would be carried out after the first operation of job 3. As soon as all the integers in the first column (from row 1 to row 6) have been considered (corresponding operations scheduled) the same process is repeated for the 2nd column and so on until all the entries in the rest of the columns are considered.

6.5.1.2 Initialization:

A stepwise procedure for the random initialization of a population of solutions can be developed as described below:

i. Initialize variables x, y and z ($x \leftarrow 0$, $y \leftarrow 0$, $z \leftarrow 0$).

ii. $i \leftarrow 1$ (where $1 \leq i \leq Jobs$)

iii. Count $\leftarrow$ 0.

iv. Generate a random number 'x' between 0 and total number of machines (*Machs*).

v. **IF** $0 \leq x < Machs$, **THEN** go to next step, **ELSE** repeat step 4.

vi. Generate another random number 'y' between 0 and total number of jobs (*Jobs*).

vii. **IF** $0 \leq y < Jobs$, **THEN** go to next step, **ELSE** repeat step 6.

viii. **IF** Pop [x, y, z] = 0, where Pop [x, y, z] is a three dimensional array for population, **THEN** Pop [x, y, z] = i, **ELSE** repeat step 4 to step 8.

ix. Increment the value of variable 'Count' (Count $\leftarrow$ Count + 1).

x. **IF** Count < *Machs*, **THEN** repeat step 4 to step 9, **ELSE** go to next step.

xi. Increment the value of i ($i \leftarrow i + 1$).

xii. **IF** $i \leq Jobs$, **THEN** repeat step 3 to step 12, **ELSE** go to next step.

xiii. Increment the value of z ($z \leftarrow z + 1$), which is a counter for the population size.

xiv. **IF** z < Pop Size (Population size), **THEN** repeat step 2 to step 14, **ELSE** go to next step.

xv. Stop.

The stepwise procedure, explained above can be further elaborated with the help of a block diagram representation as shown in Figure 6.2.

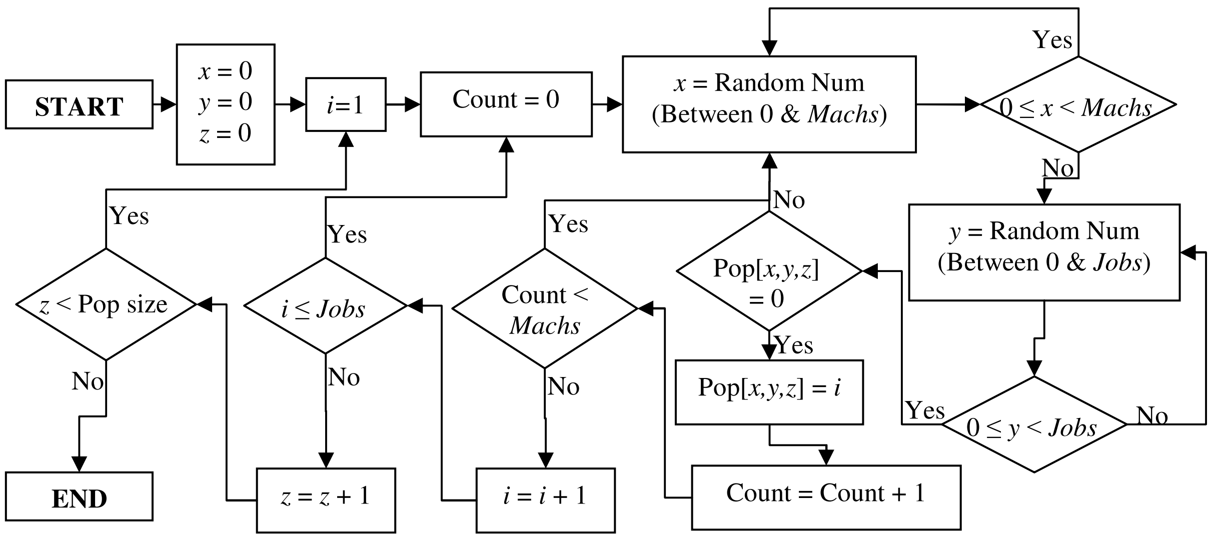


Figure 6.2: Block diagram representation of the initialization process

6.5.1.3 Evaluation and Fitness: (Tariq et al. [2007])

For the purpose of evaluation, Makespan (C_{max}) is chosen as the performance measure. It is a common performance measure and has been frequently used in literature for comparison of the quality of solutions, for different benchmark problems, developed by different techniques. Therefore, like other techniques, in this research also, each chromosome is decoded and its Makespan is calculated. The decoding procedure developed during this research is explained in the form of a stepwise procedure as follows:

i. Start by initializing the counter for rows 'i' ($i \leftarrow 0$) and the variable for Makespan 'C_{max}' ($C_{max} \leftarrow 0$).
ii. Initialize the counter for columns 'j' ($j \leftarrow 0$).
iii. Select the integer at position (i,j) ($Chrom[i,j]$).
iv. Identify the job (x), its operation number (o) and the machine (k) on which it is going to be performed.
v. **IF** it is the first operation of job (x) on machine (k) **AND** also it is the first operation of any job on machine (k), **THEN** assign a value zero to both the machine available time (MAT_k) and the job available time (JAT_x) ($MAT_k \leftarrow 0$, $JAT_x \leftarrow 0$), **ELSE** go to step 7.
vi. Assign a value zero to the earliest start time for operation 'o' ($EST_o \leftarrow 0$) **AND** go to step 8.

vii. **IF** $MAT_k < JAT_x$, **THEN** $EST_o \leftarrow MAT_k$, **ELSE** $EST_o \leftarrow JAT_x$.

viii. Calculate completion time for operation '*o*' (CT_o) by adding the earliest start time for the operation (EST_o) and it's given processing time (PT_o) ($CT_o = EST_o + PT_o$).

ix. **IF** $CT_o > C_{max}$, **THEN** $C_{max} = CT_o$, **AND** $i \leftarrow i + 1$, **ELSE** $i \leftarrow i + 1$.

x. **IF** $i < Jobs$, **THEN** repeat step 3 to step 9, **ELSE** go to next step.

xi. Increment the value of column '*j*' ($j \leftarrow j + 1$).

xii. **IF** $j < Machs$, **THEN** repeat step 2 to step 11, **ELSE** go to next step.

xiii. Makespan $\leftarrow C_{max}$.

xiv. Stop.

The procedure explained above is further elaborated in the form of a block diagram representation as in Figure 6.3.

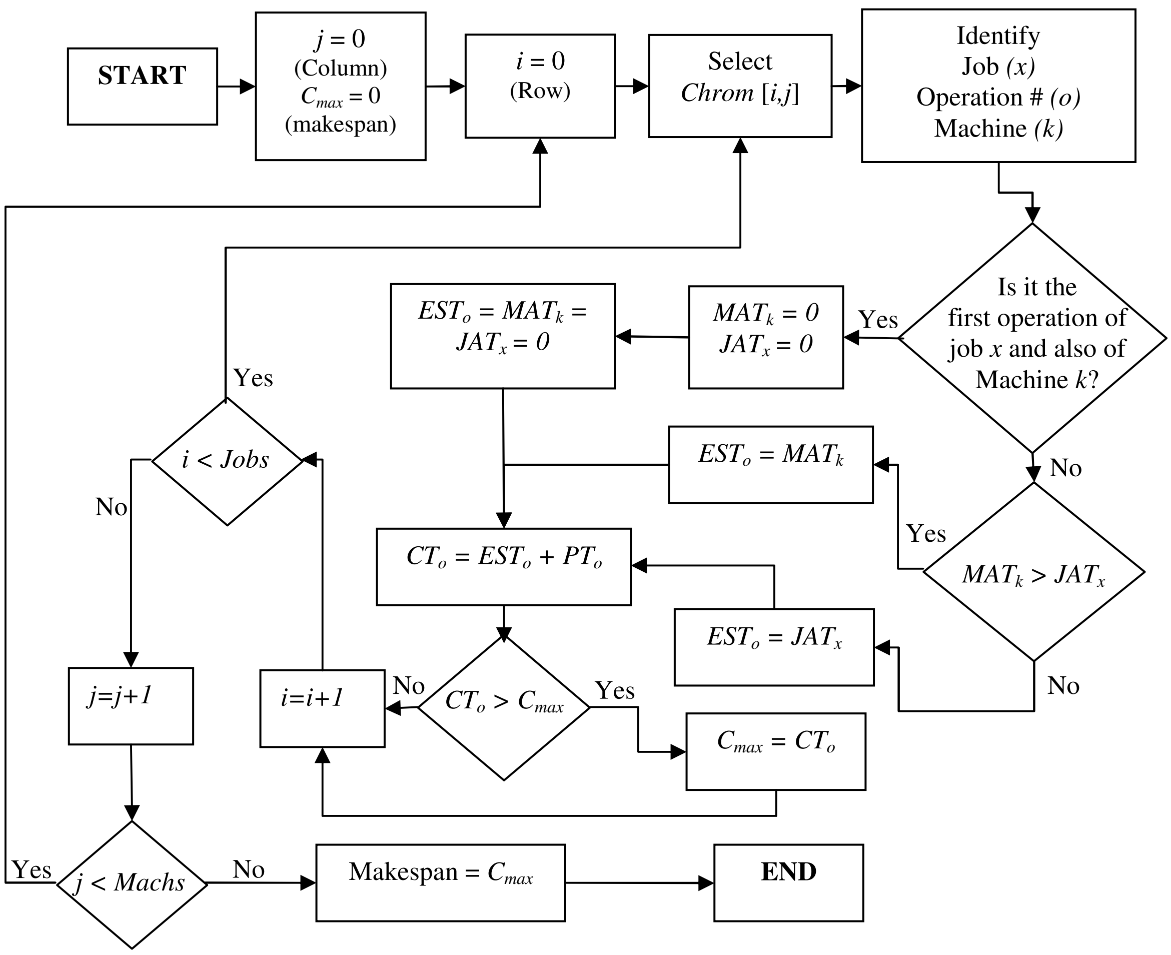


Figure 6.3: Decoding Procedure (Tariq et al. [2007])

Once the value of Makespan (C_{max}) is found then fitness value for solution can be found by taking its reciprocal as shown in Equation (6.1).

$Fitness\ Function = F = 1/C_{max}$ (6.1)

6.5.1.4 Genetic Operators: (Tariq et al. [2007])

To obtain improved results proper utilization of genetic operators (crossover and mutation) is very important. A brief description of the genetic operators used here is given in the following.

6.5.1.4.1 Crossover: (Tariq et al. [2007])

The crossover used in this particular case is a two cut point crossover, which is very common and has been previously used in literature. The stepwise procedure that is adopted to perform crossover is as follows;

i. Select the two best chromosomes ($Chrom_A$ and $Chrom_B$) of a population.
ii. Randomly select two cut points a and b, in the range from 0 to total number of jobs (*Jobs*).
iii. **IF** a < b, **THEN** proceed to next step, **ELSE** repeat step 2.
iv. Assign the value of 'a' to the counter for rows '*i*' ($i \leftarrow a$).
v. Initialize the counter for columns '*j*' ($j \leftarrow 0$).
vi. Swap the entries $Chrom_A[i,j]$ and $Chrom_B[i,j]$.
vii. Increment the value of '*j*' ($j \leftarrow j + 1$).
viii. **IF** $j < Machs$, **THEN** repeat step 6 and step 7, **ELSE** go to next step.
ix. Increment the value of '*i*' ($i \leftarrow i + 1$).
x. **IF** $i \leq b$, **THEN** repeat step 5 to step 9, **ELSE** go to next step.
xi. **IF** the crossover performed so far is less than 60% (Section 5.13) of the population size, **THEN** select the next two best chromosomes, **AND** repeat step 2 to step 10, **ELSE** go to next step.
xii. Stop.

The above stepwise procedure is also explained with the help of a block diagram representation, as shown in Figure 6.4.

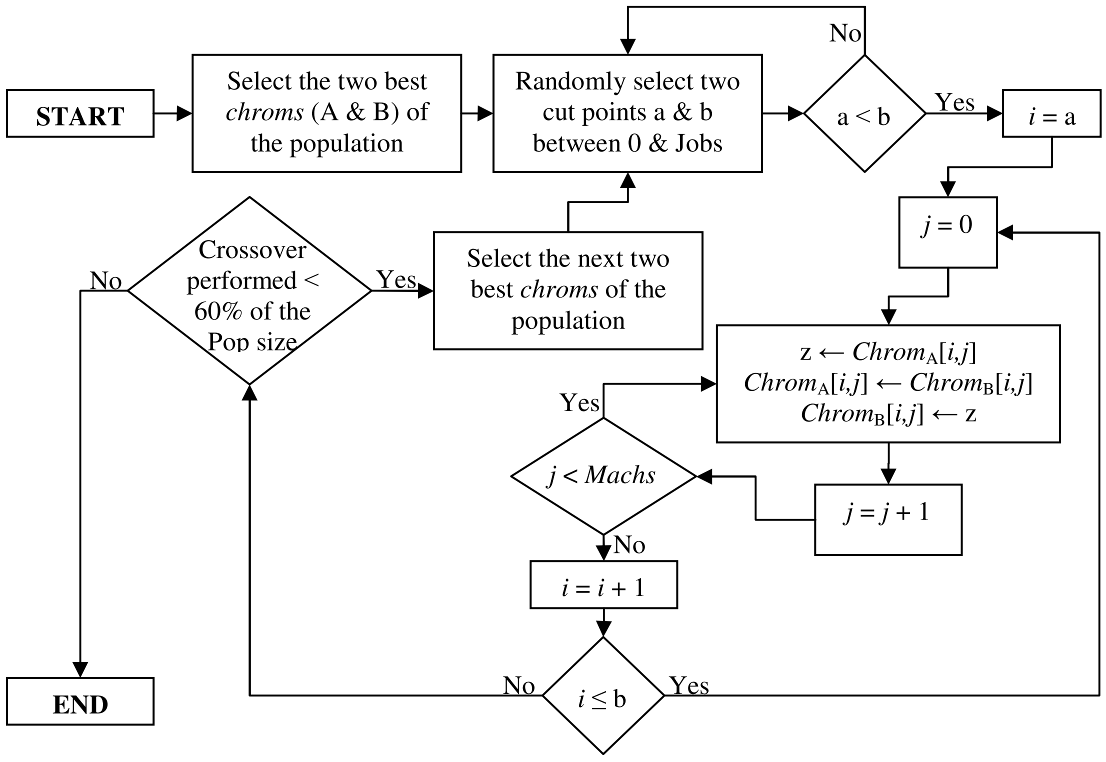

START
Select the two best *chroms* (A & B) of the population
Randomly select two cut points a & b between 0 & Jobs
No
a < b
Yes
i = a
j = 0
z ← *Chrom*A[*i,j*]
*Chrom*A[*i,j*] ← *Chrom*B[*i,j*]
*Chrom*B[*i,j*] ← z
j = *j* + 1
Yes
j < *Machs*
No
i = *i* + 1
i ≤ b
No
Yes
Select the next two best *chroms* of the population
Crossover performed < 60% of the Pop size
No
Yes
END

Figure 6.4: Block diagram representation of the crossover procedure

An example is given below to further clarify the procedure described above.

First two chromosomes are selected (Tables 6.2 & 6.3) on the basis of their better fitness (elitist strategy). Then two cut points are selected randomly. Let us say the two cut points are 3 and 4. It means the entries in rows 3 and 4 would be interchanged as shown in Tables 6.4 & 6.5. (Tariq et al. [2007])

Table 6.2: Chromosome A

3	3	1	5	4	3
4	2	1	2	5	5
1	4	6	6	6	4
2	2	5	1	5	1
6	6	4	2	2	5
4	3	3	1	6	3

Table 6.3: Chromosome B

4	5	5	1	2	2
6	1	4	4	6	6
6	4	2	6	2	5
5	4	3	3	3	1
5	5	2	2	1	1
4	3	3	3	1	6

Table 6.4: Child A

3	3	1	5	4	3
4	2	1	2	5	5
1	4	6	6	6	4
5	4	3	3	3	1
5	5	2	2	1	1
4	3	3	1	6	3

Table 6.5: Child B

4	5	5	1	2	2
6	1	4	4	6	6
6	4	2	6	2	5
2	2	5	1	5	1
6	6	4	2	2	5
4	3	3	3	1	6

The problem with this kind of crossover is that the resulting children may be of illegal nature. This illegality is because of the fact that when a portion (certain number of rows) of a chromosome is exchanged with a portion of the same size(same number of rows) of another chromosome then there is every chance that in resulting children some integers may reflect more than the specified number (number of machines) and some less than that. It has been initially mentioned that each part would have a number of operations equal to the number of machines (Section 6.5.1.1). Since in the above example (Tables 6.2 and 6.3) 6 machines and 6 parts are considered therefore in any solution an integer cannot exist less than or more than 6 times. But, if child A (Table 6.4) is examined then it can be seen that this condition is violated. As some integers exist more than 6 times (e.g. 1s = 7, 3s = 9), whereas some exist less than that (e.g. 2s = 4, 6s = 4). This shows illegality and hence some sort of repair strategy has to be developed.

6.5.1.4.2 Repair Algorithm: (Tariq et al. [2007])

As mentioned earlier the type of crossover employed may develop illegal chromosomes. Therefore, some sort of repair strategy needs to be developed to remove any illegality. For this purpose a repair procedure is developed during this research. First each child is checked for legality. This can be done by counting the number of entries each integer has in a child. If the number of entries of any one or more integers is less than or greater than the number of machines then such a chromosome is called illegal. If found illegal, then repair is carried out, otherwise the repair strategy is skipped. Once illegality is confirmed, then information about the positions of those integers which are in excess and those integers which are in shortage is stored. Remember only those positions are stored which are outside the portion which took part in crossover. Now one by one those integers are selected which are in shortage and then randomly any of the stored positions is picked and assigned the value equal to the selected integer. This process is repeated until all the integers which are in shortage are placed back into the solution/child.

The procedure developed during this research for repairing illegal solutions is mainly divided into two main portions. In the first portion it is determined whether a child is legal or not, whereas the second portion deals with repair of an illegal child.

Both the portions have been separately represented with the help of block diagrams in Figures 6.5 and 6.6.

A stepwise procedure for the legality check of a Child [*i,j*] is as follows:

i. Start with an initial value of a variable 'Gene' i.e. 1 (Gene ← 1)
ii. Initialize variables 'Count' (counter for counting the number of entries of each integer) and '*i*' (counter for rows) (Count ← 0, and *i* ← 0).
iii. Initialize '*j*' (counter for columns) (*j* ← 0).
iv. **IF** Child [*i,j*] = Gene, **THEN** Count ← Count +1, **AND** *j* ← *j* + 1, **ELSE** *j* ← *j* + 1.
v. **IF** *j* < *Machs*, **THEN** repeat step 4, **ELSE** go to next step.
vi. Increment the counter for rows '*i*' (*i* ← *i* + 1)
vii. **IF** *i* < *Jobs*, **THEN** repeat steps 3 to step 6, **ELSE** go to next step.
viii. **IF** Count = *Machs*, **THEN** Gene ← Gene + 1, **AND** go to next step, **ELSE** stop as the illegality of child is proved.
ix. Increment the value of 'Gene' (Gene ← Gene + 1)
x. **IF** Gene ≤ *Machs*, **THEN** repeat step 2 to step 9, **ELSE** stop with the conclusion that child is legal and doesn't require any repair.

The working of the above described stepwise procedure for legality check is further explained in the form of a block diagram representation in Figure 6.5.

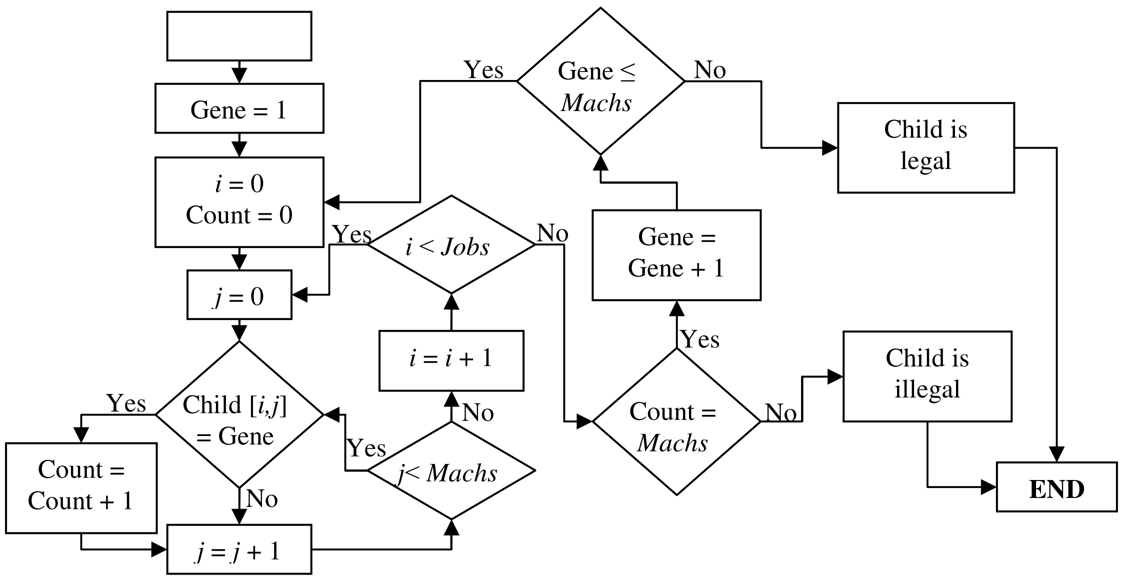


Figure 6.5: Block diagram representation for legality check

Once it is confirmed that a child is illegal then the repair procedure, developed during this research, is applied. The sequence of steps is described below to carry out repair of an illegal solution.

i. Start with a minimum value of 1 for a variable ‘Gene’ (Gene $\leftarrow$ 1), **AND** initialize counters ‘Count, *z, w*’ (Count $\leftarrow$ 0, $z \leftarrow 0$, $w \leftarrow 0$).
ii. Initialize a variable ‘Num’ (Num $\leftarrow$ 0) that is used to count the number of entries of each integer in a solution.
iii. Initialize counter for columns ‘*j*’ ($j \leftarrow 0$).
iv. Initialize counter for rows ‘*i*’ ($i \leftarrow 0$).
v. **IF** Child$_A[i,j]$ = Gene, **THEN** increment the value of ‘Num’ (Num $\leftarrow$ Num + 1), **AND** increment the value of the variable for rows ‘*i*’ ($i \leftarrow i + 1$), **ELSE** increment ‘*i*’ ($i \leftarrow i + 1$).
vi. **IF** $i < Jobs$, **THEN** repeat step 5, **ELSE** increment the value of variable for columns ‘*j*’ ($j \leftarrow j + 1$).
vii. **IF** $j < Machs$, **THEN** repeat step 4 to step 6, **ELSE** go to next step.
viii. **IF** Num = *Machs*, **THEN** increment the value of ‘Gene’ (Gene $\leftarrow$ Gene + 1), **AND** go to next step, **ELSE** go to step 10.
ix. **IF** Gene $\leq$ *Machs*, **THEN** repeat step 2 to step 8, **ELSE** go to step 13.
x. **IF** Num < *Machs*, **THEN** store the integer along side those integers which are in shortage (Short [z] $\leftarrow$ Gene) **AND** its shortage amount into another array (S. Amnt [z] $\leftarrow$ *Machs* – Num) **AND** Count $\leftarrow$ Count + S. Amnt [z] **AND** increment ‘*z*’ ($z \leftarrow z + 1$), **ELSE** store that integer into another array allocated for those integers which are in excess (Excess[w] $\leftarrow$ Gene) **AND** its excess amount into its corresponding array (E. Amnt [w] $\leftarrow$ Num – *Machs*) **AND** increment ‘*w*’ ($w \leftarrow w + 1$).
xi. Increment the value of variable ‘Gene’ (Gene $\leftarrow$ Gene + 1).
xii. **IF** Gene $\leq$ *Machs*, **THEN** repeat step 2 to step 11, **ELSE** go to next step.
xiii. $X_1 \leftarrow$ crossover cut point # 1, $X_2 \leftarrow$ crossover cut point # 2.
xiv. Initialize variables ‘Num, *z, w*’ (Num $\leftarrow$ 0, $z \leftarrow 0$, $w \leftarrow 0$).
xv. Generate two random numbers x (between 0 and *Jobs*) and y (between 0 and *Machs*).
xvi. **IF** $X_1 \leq x \leq X_2$, **THEN** repeat step 15, **ELSE** go to next step.
xvii. **IF** Child$_A[x,y]$ = Excess[w], **THEN** go to next step, **ELSE** repeat step 15 and step 16.

xviii. Child$_A[x,y]$ ← Short $[z]$ **AND** S. Amnt$[z]$ ← S. Amnt$[z]$ – 1.

xix. **IF** S. Amnt$[z]$ = 0, **THEN** $z \leftarrow z + 1$ **AND** E. Amnt$[w]$ ← E. Amnt$[w]$ – 1, **ELSE** E. Amnt$[w]$ ← E. Amnt$[w]$ – 1.

xx. **IF** E. Amnt$[w]$ = 0, **THEN** $w \leftarrow w + 1$ **AND** Num ← Num + 1, **ELSE** Num ← Num + 1.

xxi. **IF** Num < Count, **THEN** repeat step 15 to step 20, **ELSE** go to next step.

xxii. Stop.

The above described procedure can also be expressed in the form of block diagram representation, as shown in Figure 6.6.

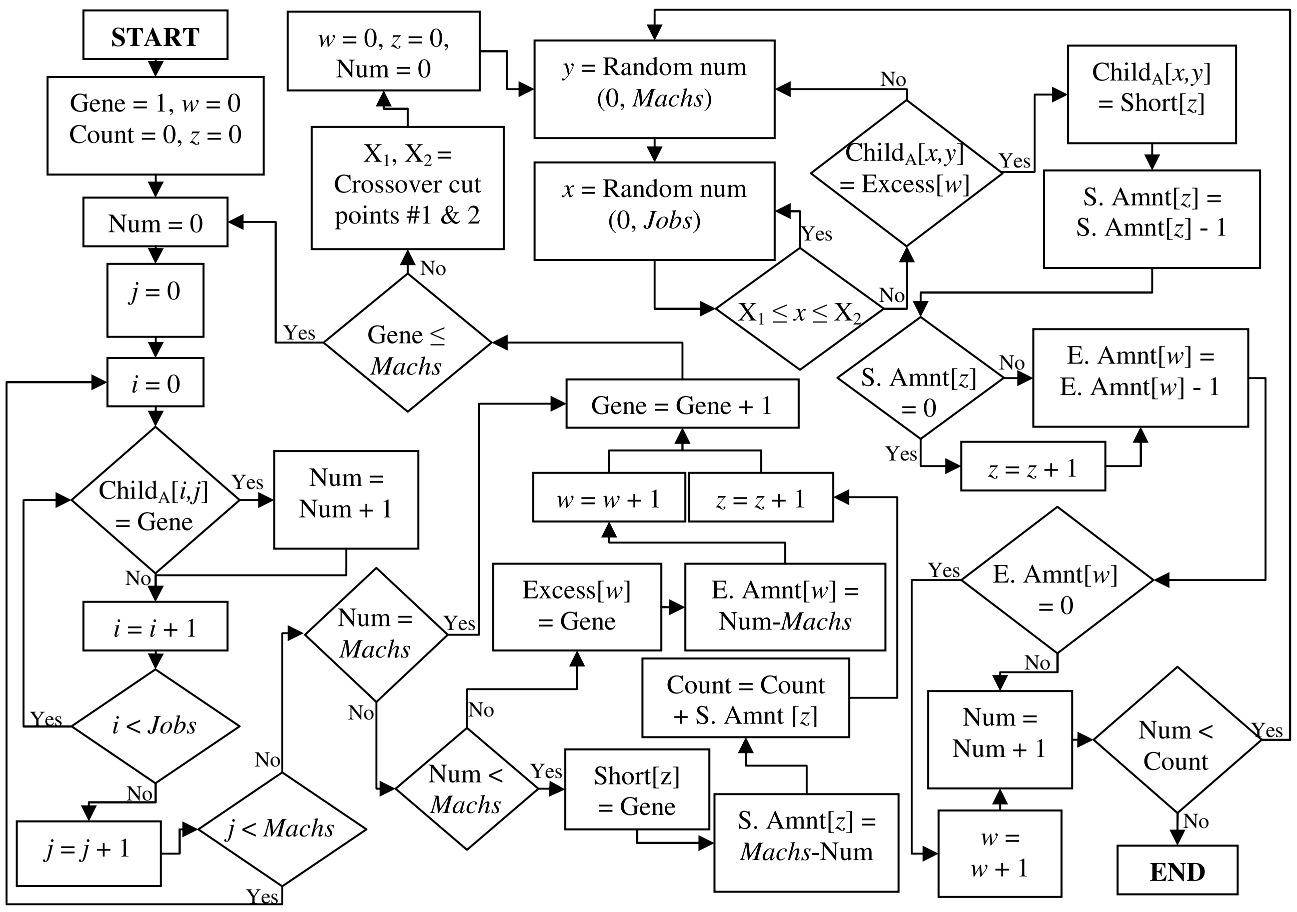


Figure 6.6: Block diagram representation for repair algorithm

In order to explain the repair algorithm being proposed here in more detail, child "A" (Table 6.4) is subjected to it.

Step 1: First the solution is checked for legality as shown in Table 6.6:

Table 6.6: Number of times each integer exists in the solution

Integer	Existence (How many times?)	Short/ Excess/ Legal	Num of times in excess/shortage
1	7	Excess	1
2	4	Short	2
3	9	Excess	3
4	6	Legal	0
5	6	Legal	0
6	4	Short	2

Step 2: As the frequency of existence of some integers exceeds (excess) the limit (total number of machines) and in some cases it is less than (short), therefore it shows that the solution is not legal and repair is required.

Step 3: Now we are aware about the number of times each integer exists in the solution i.e. about their excess and shortage (if any). So, information about the locations of those integers which exceed the limit is stored. Here one thing is important to mention that only those locations are considered for storage which are on the outer side of the portion which has been crossed over (rows 3 & 4). The relevant information is presented in Table 6.7.

Table 6.7: Positions of those integers which are in excess

Integers in excess	Positions	
	x (Row)	y (Column)
1	0	2
	1	2
	2	0
	5	3
3	0	0
	0	1
	0	5
	5	1
	5	2
	5	5

Step 4: Now one by one those integers which are in shortage are selected and placed randomly in any of the previously stored positions as shown in Table 6.8.

Table 6.8: Repair work

Integer in shortage	Shortage amount	Integers in excess	Excess amount	Randomly selected positions		New value assigned to selected position
				x(row)	*y(col)*	
2	2	1	1	1	2	2
		3	3	0	0	2
6	2			0	5	6
				5	5	6

Step 5: It is evident from Table 5.19, that step 4 has been repeated until all the integers, in shortage, are being placed in the solution. The repaired solution is shown in Table 6.9.

Table 6.9: Repaired solution

2	**3**	**1**	**5**	**4**	**6**
4	**2**	**2**	**2**	**5**	**5**
1	**4**	**6**	**6**	**6**	**4**
5	**4**	**3**	**3**	**3**	**1**
5	**5**	**2**	**2**	**1**	**1**
4	**3**	**3**	**1**	**6**	**6**

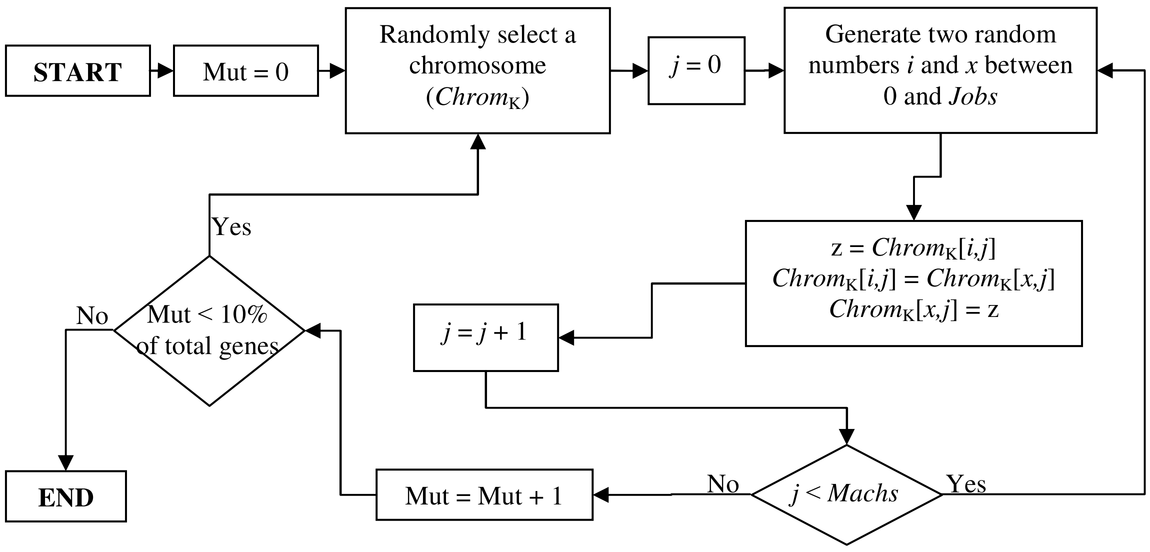
6.5.1.4.3 Mutation (Tariq et al. [2007])

In this case swap mutation is used which has been frequently used in literature. Here, the values of two randomly selected genes are swapped. The process is continued until 10% randomly selected genes of the total number of genes in population get mutated (Section 5.13). This process is further described in a stepwise manner as follows:

i. Start with an initialized value of a variable for mutation 'Mut' (Mut ← 0).
ii. Randomly select a chromosome '$Chrom_K$' from the population.
iii. Initialize the variable for columns 'j' ($j \leftarrow 0$).
iv. Generate two random numbers 'i & x' between 0 and *Jobs*.
v. Swap the entries $Chrom_K[i,j]$ and $Chrom_K[x,j]$.
vi. Increment the value of 'j' ($j \leftarrow j + 1$).
vii. **IF** $j < Machs$, **THEN** repeat step 4 to step 6, **ELSE** go to next step.
viii. Increment the value of 'Mut' (Mut ← Mut + 1).
ix. **IF** Mut < 10% of total genes in population (Section 5.13), **THEN** repeat step 2 to step 8, **ELSE** go to next step.

x. Stop.

The stepwise procedure for mutation described above can also be elaborated in the form of a block diagram representation as follows in Figure 6.7.



START
Mut = 0
Randomly select a chromosome ($Chrom_K$)
$j = 0$
Generate two random numbers i and x between 0 and $Jobs$
$z = Chrom_K[i,j]$
$Chrom_K[i,j] = Chrom_K[x,j]$
$Chrom_K[x,j] = z$
$j = j + 1$
$j < Machs$
Yes
No
Mut = Mut + 1
Mut < 10% of total genes
Yes
No
END

Figure 6.7: Block diagram representation of mutation process

Mutation performed in this way guarantees to generate legal offspring as shown in Tables 6.10 & 6.11. (Tariq et al. [2007])

Table 6.10: Chrom selected

3	3	1	5	4	3
4	2	1	2	5	5
1	4	6	6	6	4
5	4	3	3	3	1
5	5	2	2	1	1
4	3	3	1	6	3

Table 6.11: Mutated chromosome

3	3	1	2	6	3
4	5	1	2	5	5
5	4	3	6	4	1
1	4	3	3	3	4
5	2	2	5	1	1
4	3	6	1	6	3

6.5.1.4.4 Selection: (Tariq et al. [2007])

A number of selection procedures are available in literature that can be used to select a new population from the existing one. All these selection procedures are actually based on inspiration from the Darwin's Evolution Theory. In roulette wheel selection procedure, which is very common and has been frequently used in literature, the selection of a chromosome into the next generation depends mainly on its fitness value. Therefore, a chromosome with a higher fitness value can get selected more than once into the next generation thus reducing the diversity of the population. The frequency of this behaviour (selection of multiple copies of one chromosome) would increase in the coming generations as more and more similar solutions would get selected which may lead the algorithm to a premature convergence. On the other hand the method used for selection of new population is known as Stochastic Universal Sampling (SUS), presented by Chaperfield et al. [2001] and also used by Pohlheim [2005], used here for the process of selecting chromosomes in the next generation from the present generation. This method is used due to its nature of having minimum spread and displaying bias as zero. To clarify it further let us consider an example in which there are 10 solutions/chromosomes in a population. The fitness values, selection probabilities and cumulative probabilities of all the chromosomes and the rest of the procedure is as follows:

Fitness values: (the fitness function values for each chromosome in the population)

F = 0.81, 0.24, 0.56, 0.72, 0.31, 0.43, 0.54, 0.26, 0.62, 0.45

Fitness sum: (the total sum of the fitness values of all the chromosomes in the population)

∑F = 4.94

Selection probability (S. Prob.): (obtained by dividing the individual fitness value of a chromosome in a population by the total fitness value of the population)

S. Prob. = F/ (∑F) = 0.164, 0.049, 0.113, 0.146, 0.063, 0.087, 0.11, 0.053, 0.126, 0.091

Cumulative probability (C. Prob.): (obtained by continuously adding the fitness values for each chromosome. For example cumulative probability for first chromosome is its own fitness value whereas for the second chromosome it would be the sum of the fitness values of first two chromosomes, for the third chromosome it would be the sum of the fitness values of first three chromosomes and so on)

C. Prob. = 0.164, 0.213, 0.326, 0.472, 0.535, 0.622, 0.732, 0.785, 0.838, 1.00

Once the cumulative probabilities for all the chromosomes are calculated then a random number is generated between 0 and $1/z$, where z represents the population size. For the above example the value of $1/z$ is 0.1 as there are 10 chromosomes in the population. Now a random number is generated in the range (0, 0.1). For example the number randomly generated is 0.09. Since it is less than the cumulative probability of first chromosome (0.164) therefore first chromosome is selected into the next generation. For the selection of second chromosome the value of randomly generated number is doubled. Since the doubled value (0.18) is between the cumulative probability values of first and second chromosome, therefore second chromosome is selected. Similarly, for the selection of third chromosome the value of the randomly generated number is tripled and then compared with the cumulative probability values and a respective chromosome is selected. This process is kept continued until all the members of the next generation are selected.

The procedure shows that the chances of selection of multiple copies of one chromosome are less as compared to roulette wheel selection approach thus maintaining a reasonable diversity in population in each generation.

This method can be expressed in the form of a sequence of steps as follows:

i. Start by initializing the values of a variable 'Sum' and a counter 'z' (Sum $\leftarrow 0$, $z \leftarrow 0$).
ii. Sum $\leftarrow$ Sum + F. Value [z], where 'F. Value [z]' is the fitness value of chromosome at position z ($Chrom_z$) in the array of population.
iii. Increment the value of 'z' ($z \leftarrow z + 1$).
iv. **IF** $z <$ Pop Size, **THEN** repeat step 2 and step 3, **ELSE** go to next step.
v. Initialize the counter 'z' ($z \leftarrow 0$).
vi. Calculate selection probability for $Chrom_z$ (Sel. Prob [z] = F. Value [z] / Sum).
vii. Increment the value of 'z' ($z \leftarrow z + 1$).
viii. **IF** $z <$ Pop Size, **THEN** repeat step 6 and step 7, **ELSE** go to next step.
ix. Sum $\leftarrow 0$ **AND** $z \leftarrow 0$.
x. Sum $\leftarrow$ Sum + Sel. Prob [z].
xi. Assign the value of Sum to the cumulative probability of $Chrom_z$ (Cum Prob [z] $\leftarrow$ Sum)

xii. Increment the value of 'z' ($z \leftarrow z + 1$).

xiii. **IF** $z <$ Pop Size, **THEN** repeat step 10 to step 12, **ELSE** go to next step.

xiv. Initialize the pointer 'R' and a variable 'K' (R ← 0 **AND** K ← 0).

xv. Generate a random number for pointer 'R' between 0 and 1/ Pop Size.

xvi. Initialize a variable 'x' ($x \leftarrow 0$).

xvii. **IF** R ≤ Cum Prob [x], **THEN** select $Chrom_x$, **ELSE IF** R ≤ Cum Prob [x + 1], **THEN** select $Chrom_{x+1}$, **ELSE** $x \leftarrow x + 1$ **AND** repeat step 17.

xviii. Increment the value of 'K' (K ← K + 1).

xix. **IF** K < Pop Size, **THEN** R ← R + 1/ Pop Size **AND** repeat step 17 and step 18, **ELSE** go to next step.

xx. Stop.

A block diagram representation for the above stepwise procedure can be as shown in Figure 6.8.

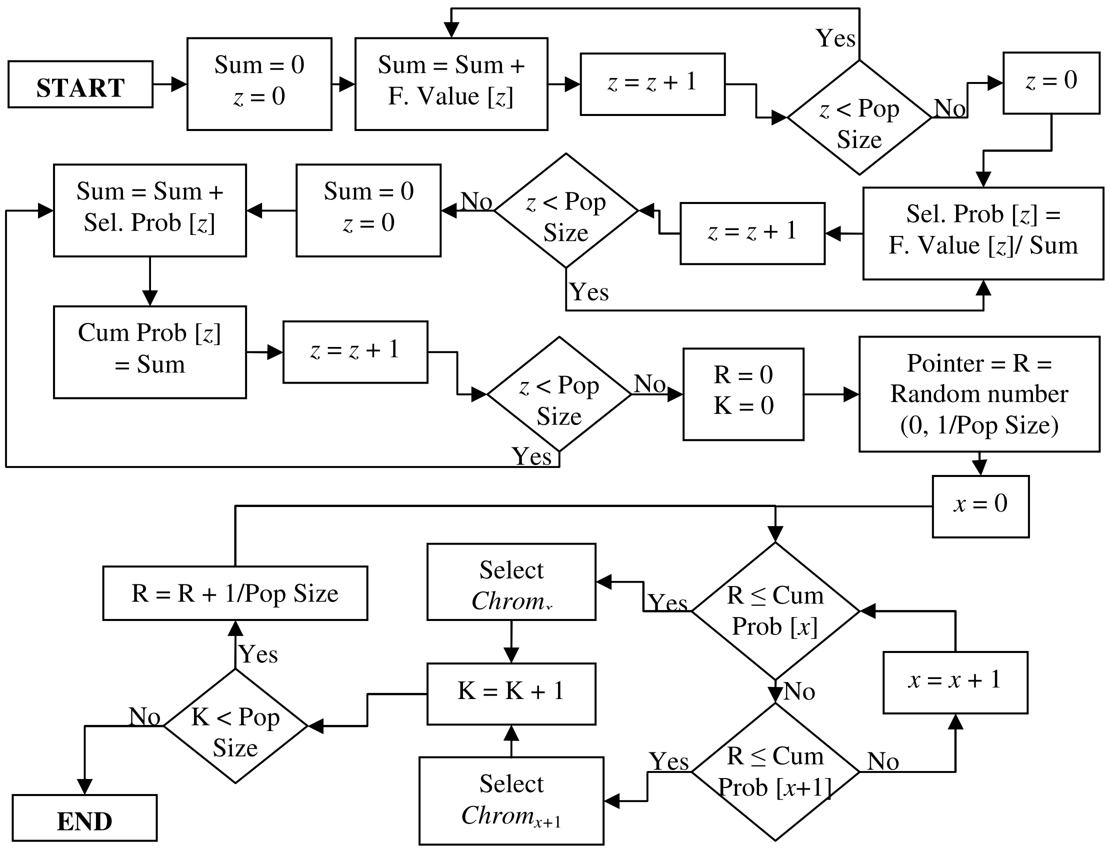


Figure 6.8: Stochastic Universal Sampling (SUS)

6.5.2 Local Search Heuristic (LSH): (Tariq et al. [2007])

The LSH, developed during this research, is such that it carries out the local improvement of the best solution in each generation. The LSH is started by swapping the first two entries (entry in first column and first row with entry in second column and first row) in the solution that is undergoing local improvement. If this swapping causes a decrease in the actual Makespan or it remains the same then this change is stored, else the previous arrangement is restored. After this the position of the same gene is swapped with the next gene which is in first row and third column and so on. Once swapping with all the genes in first row has been carried out then the same procedure for the same gene is kept continued in the second row and so on until all the entries in a solution are swapped with the first gene and corresponding Makespan values calculated. Once the testing of a particular gene with all the genes in a solution is completed then the same is repeated for the next gene and this process kept continued for 50% of the $Machs \times Jobs$ times. The justification for this limit ($50\% \times Machs \times Jobs$ times) is that because of the fact that once first half of the operations are scheduled then by that time a pattern has been developed and therefore the scheduling of the last half of the operations does not affect the value of the Makespan of the problem. Relevant details are given in the following and complete procedure of local improvement for one gene is shown in Table 6.14.

In spite of the fact that LSH displayed considerable effectiveness; it still needed the abilities of GA as a far as searching a better solution is concerned which can be further transformed into the best. The LSH, developed during this research, is clearly explained with the help of a stepwise procedure as described below:

i. Start by initializing a counter 'Count' (Count $\leftarrow$ 0).
ii. Initialize counters for rows 'a' and columns 'b' ($a \leftarrow 0$ **AND** $b \leftarrow 0$).
iii. Initialize counters for rows 'i' and columns 'j' ($i \leftarrow 0$ **AND** $j \leftarrow 0$).
iv. **IF** Best [i,j] = Best [a,b], **THEN** $b \leftarrow b + 1$ **AND** go to step 9, **ELSE** go to next step.
v. Swap the entries Best [i,j] and Best [a,b].
vi. Decode the solution and calculate its Makespan 'C'.
vii. **IF** $C < C_{max}$, where C_{max} is the initial Makespan of the solution, **THEN** store the change **AND** $i \leftarrow a, j \leftarrow b$, **ELSE** reverse the swapped entries.

viii. Increment the value of 'b' ($b \leftarrow b + 1$).

ix. **IF** $b < Machs$, **THEN** repeat step 4 to step 8, **ELSE** go to next step.

x. $a \leftarrow a + 1$ **AND** $b \leftarrow 0$.

xi. **IF** $a < Jobs$, **THEN** repeat step 4 to step 10, **ELSE** go to next step.

xii. Count ← Count + 1.

xiii. **IF** Count < ½ (*Machs* × *Jobs*), **THEN** repeat step 2 to step 12, **ELSE** go to next step.

xiv. Stop.

This stepwise procedure can also be expressed in the form of a block diagram representation as shown in Figure 6.9.

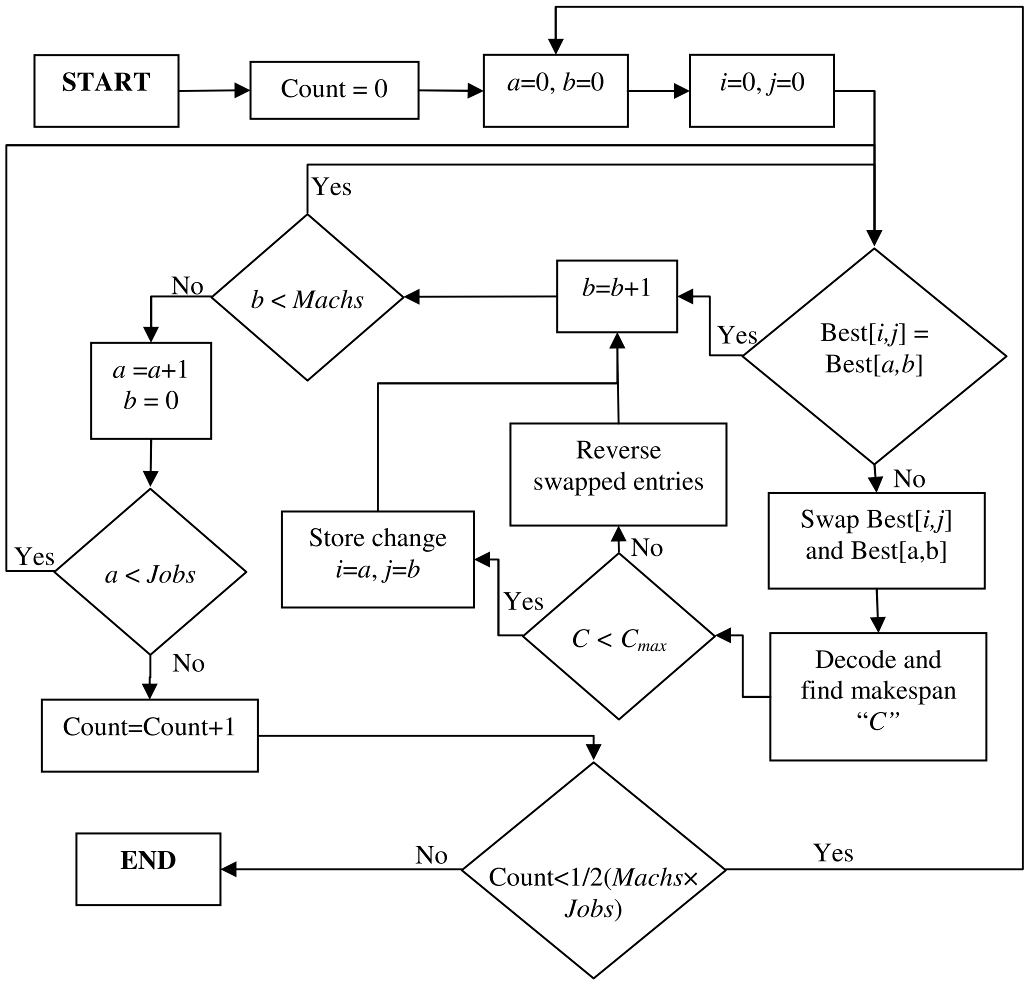


Figure 6.9: Local Search Heuristic (LSH) (Tariq et al. [2007])

To explain the procedure of LSH, developed during this research, a numerical example of order 4×4 is generated randomly as shown in Table 6.12.

Table 6.12: A randomly generated 4×4 problem

Jobs	Operations (Machine, time)			
	1	2	3	4
	(m, t)	(m, t)	(m, t)	(m, t)
1	3,5	2,7	4,6	1,4
2	2,6	1,8	3,7	4,5
3	4,9	3,3	1,4	2,2
4	1,5	4,8	2,6	3,3

The problem is required to be solved with the objective of minimizing the Makespan (C_{max}). to accomplish this objective the data of the problem is fed into a computer code developed in accordance with the methodology explained in Figure 5.17. In its first generation GA developed the following best solution (Table 6.13).

Table 6.13: Best chromosome after 1st generation of GA

2	2	2	4
4	3	4	3
3	4	1	2
1	1	1	3

The Makespan of the solution presented in Table 6.13= C_{max} = 33. After being decoded the Gantt Chart representation of the solution is as shown in Figure 6.10.

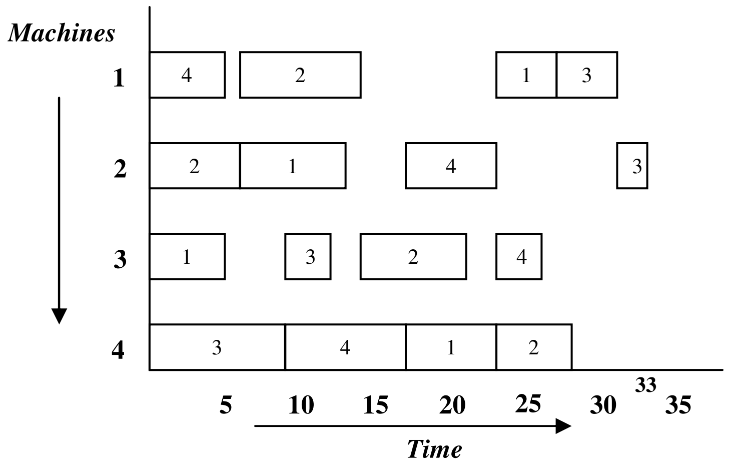


Figure 6.10: Schedule developed for the best chromosome found in 1st generation of GA

The best solution (Table 6.13) presented by GA in its first generation is now further locally improved with the help of LSH. For the first gene of the solution the complete LSH procedure is as shown in Table 6.14.

Table 6.14: Complete stepwise procedure of local improvement for one gene

Solution Showing selected genes	Selected Genes are equal?	Interchange of selected genes	*C*	Comparison of *C* and C_{max}	Remarks	Solution after the change is validated
$\begin{array}{\|c\|c\|c\|c\|}\hline \boxed{2} & \boxed{2} & 2 & 4 \\ \hline 4 & 3 & 4 & 3 \\ \hline 3 & 4 & 1 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$	Yes	-	-	-	Since genes are equal so move to next gene	$\begin{array}{\|c\|c\|c\|c\|}\hline 2 & 2 & 2 & 4 \\ \hline 4 & 3 & 4 & 3 \\ \hline 3 & 4 & 1 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$
$\begin{array}{\|c\|c\|c\|c\|}\hline \boxed{2} & 2 & \boxed{2} & 4 \\ \hline 4 & 3 & 4 & 3 \\ \hline 3 & 4 & 1 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$	Yes	-	-	-	Since genes are equal so move to next gene	$\begin{array}{\|c\|c\|c\|c\|}\hline 2 & 2 & 2 & 4 \\ \hline 4 & 3 & 4 & 3 \\ \hline 3 & 4 & 1 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$
$\begin{array}{\|c\|c\|c\|c\|}\hline \boxed{2} & 2 & 2 & \boxed{4} \\ \hline 4 & 3 & 4 & 3 \\ \hline 3 & 4 & 1 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$	No	$\begin{array}{\|c\|c\|c\|c\|}\hline \boxed{4} & 2 & 2 & \boxed{2} \\ \hline 4 & 3 & 4 & 3 \\ \hline 3 & 4 & 1 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$	42	$C_{max} < C$	No improvement so change is reverted	$\begin{array}{\|c\|c\|c\|c\|}\hline 2 & 2 & 2 & 4 \\ \hline 4 & 3 & 4 & 3 \\ \hline 3 & 4 & 1 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$
$\begin{array}{\|c\|c\|c\|c\|}\hline \boxed{2} & 2 & 2 & 4 \\ \hline \boxed{4} & 3 & 4 & 3 \\ \hline 3 & 4 & 1 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$	No	$\begin{array}{\|c\|c\|c\|c\|}\hline \boxed{4} & 2 & 2 & 4 \\ \hline \boxed{2} & 3 & 4 & 3 \\ \hline 3 & 4 & 1 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$	33	$C_{max} = C$	Change is stored	$\begin{array}{\|c\|c\|c\|c\|}\hline 4 & 2 & 2 & 4 \\ \hline 2 & 3 & 4 & 3 \\ \hline 3 & 4 & 1 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$
$\begin{array}{\|c\|c\|c\|c\|}\hline 4 & 2 & 2 & 4 \\ \hline \boxed{2} & \boxed{3} & 4 & 3 \\ \hline 3 & 4 & 1 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$	No	$\begin{array}{\|c\|c\|c\|c\|}\hline 4 & 2 & 2 & 4 \\ \hline \boxed{3} & \boxed{2} & 4 & 3 \\ \hline 3 & 4 & 1 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$	40	$C_{max} < C$	No improvement so change is reverted	$\begin{array}{\|c\|c\|c\|c\|}\hline 4 & 2 & 2 & 4 \\ \hline 2 & 3 & 4 & 3 \\ \hline 3 & 4 & 1 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$
$\begin{array}{\|c\|c\|c\|c\|}\hline 4 & 2 & 2 & 4 \\ \hline \boxed{2} & 3 & \boxed{4} & 3 \\ \hline 3 & 4 & 1 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$	No	$\begin{array}{\|c\|c\|c\|c\|}\hline 4 & 2 & 2 & 4 \\ \hline \boxed{4} & 3 & \boxed{2} & 3 \\ \hline 3 & 4 & 1 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$	42	$C_{max} < C$	No improvement so change is reverted	$\begin{array}{\|c\|c\|c\|c\|}\hline 4 & 2 & 2 & 4 \\ \hline 2 & 3 & 4 & 3 \\ \hline 3 & 4 & 1 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$
$\begin{array}{\|c\|c\|c\|c\|}\hline 4 & 2 & 2 & 4 \\ \hline \boxed{2} & 3 & 4 & \boxed{3} \\ \hline 3 & 4 & 1 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$	No	$\begin{array}{\|c\|c\|c\|c\|}\hline 4 & 2 & 2 & 4 \\ \hline \boxed{3} & 3 & 4 & \boxed{2} \\ \hline 3 & 4 & 1 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$	45	$C_{max} < C$	No improvement so change is reverted	$\begin{array}{\|c\|c\|c\|c\|}\hline 4 & 2 & 2 & 4 \\ \hline 2 & 3 & 4 & 3 \\ \hline 3 & 4 & 1 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$
$\begin{array}{\|c\|c\|c\|c\|}\hline 4 & 2 & 2 & 4 \\ \hline \boxed{2} & 3 & 4 & 3 \\ \hline \boxed{3} & 4 & 1 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$	No	$\begin{array}{\|c\|c\|c\|c\|}\hline 4 & 2 & 2 & 4 \\ \hline \boxed{3} & 3 & 4 & 3 \\ \hline \boxed{2} & 4 & 1 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$	33	$C_{max} = C$	Change is stored	$\begin{array}{\|c\|c\|c\|c\|}\hline 4 & 2 & 2 & 4 \\ \hline 3 & 3 & 4 & 3 \\ \hline 2 & 4 & 1 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$
$\begin{array}{\|c\|c\|c\|c\|}\hline 4 & 2 & 2 & 4 \\ \hline 3 & 3 & 4 & 3 \\ \hline \boxed{2} & \boxed{4} & 1 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$	No	$\begin{array}{\|c\|c\|c\|c\|}\hline 4 & 2 & 2 & 4 \\ \hline 3 & 3 & 4 & 3 \\ \hline \boxed{4} & \boxed{2} & 1 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$	33	$C_{max} = C$	Change is stored	$\begin{array}{\|c\|c\|c\|c\|}\hline 4 & 2 & 2 & 4 \\ \hline 3 & 3 & 4 & 3 \\ \hline 4 & 2 & 1 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$
$\begin{array}{\|c\|c\|c\|c\|}\hline 4 & 2 & 2 & 4 \\ \hline 3 & 3 & 4 & 3 \\ \hline 4 & \boxed{2} & \boxed{1} & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$	No	$\begin{array}{\|c\|c\|c\|c\|}\hline 4 & 2 & 2 & 4 \\ \hline 3 & 3 & 4 & 3 \\ \hline 4 & \boxed{1} & \boxed{2} & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$	33	$C_{max} = C$	Change is stored	$\begin{array}{\|c\|c\|c\|c\|}\hline 4 & 2 & 2 & 4 \\ \hline 3 & 3 & 4 & 3 \\ \hline 4 & 1 & 2 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$
$\begin{array}{\|c\|c\|c\|c\|}\hline 4 & 2 & 2 & 4 \\ \hline 3 & 3 & 4 & 3 \\ \hline 4 & 1 & \boxed{2} & \boxed{2} \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$	Yes	-	-	-	Since genes are equal so move to next gene	$\begin{array}{\|c\|c\|c\|c\|}\hline 4 & 2 & 2 & 4 \\ \hline 3 & 3 & 4 & 3 \\ \hline 4 & 1 & 2 & 2 \\ \hline 1 & 1 & 1 & 3 \\ \hline\end{array}$

4 2 2 4 3 3 4 3 4 1 2 2 1 1 1 3	No	4 2 2 4 3 3 4 3 4 1 1 2 2 1 1 3	40	$C_{max} < C$	No improvement so change is reverted	4 2 2 4 3 3 4 3 4 1 2 2 1 1 1 3
4 2 2 4 3 3 4 3 4 1 2 2 1 1 1 3	No	4 2 2 4 3 3 4 3 4 1 1 2 1 2 1 3	33	$C_{max} = C$	Change is stored	4 2 2 4 3 3 4 3 4 1 1 2 1 2 1 3
4 2 2 4 3 3 4 3 4 1 1 2 1 2 1 3	No	4 2 2 4 3 3 4 3 4 1 1 2 1 1 2 3	33	$C_{max} = C$	Change is stored	4 2 2 4 3 3 4 3 4 1 1 2 1 1 2 3
4 2 2 4 3 3 4 3 4 1 1 2 1 1 2 3	No	4 2 2 4 3 3 4 3 4 1 1 2 1 1 3 2	38	$C_{max} < C$	No improvement so change is reverted	4 2 2 4 3 3 4 3 4 1 1 2 1 1 2 3

Once process for the first gene is completed it is restarted, for the new first gene, all over again and the same is kept continued for 50%×*Machs*×*Jobs,* times. After the procedure is completed the solution locally improved by LSH is presented Table 6.15:

Table 6.15: Solution after being locally improved

2	1	3	1
3	1	3	2
4	4	1	4
4	2	3	2

Makespan = $C_{max} = 28$

The solution shown in Table 6.15 can be represented on a Gantt chart as shown in Figure 6.11, with a Makespan of 28 (17.85% improvement).

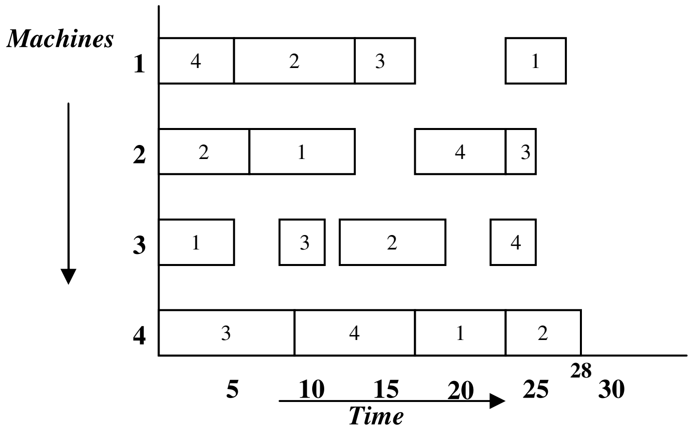


Figure 6.11: Schedule developed for the solution improved by LSH

6.6 Numerical Example of Job-Shop Scheduling: (Tariq et al. [2007])

To further justify the effectiveness of the approach developed for JSSP during this research a benchmark numerical example (Fisher & Thompson [1963]) is solved here. Information about the processing sequence and corresponding processing times is given in Table 6.16.

A legal schedule can be defined as that schedule in which the sequence, in which a job is to be processed on different machines, is to be maintained. This actually means that there must not be two operations scheduled on one machine at one time, similarly two operations of one job must not be scheduled on two different machines at one time.

Table 6.16: A 6 × 6 bench mark problem [Fisher & Thomson (1963)]

	Operations (Time, Machine)					
	1	**2**	**3**	**4**	**5**	**6**
Jobs	**(t, m)**	**(t, m)**	**(t, m)**	**(t, m)**	**(t, m)**	**(t, m)**
1	1,3	3,1	6,2	4,7	3,6	6,5
2	8,2	5.3	10,5	6,10	10,1	4,4
3	5,3	4,4	8,6	1,9	1,2	7,5
4	5,2	5,1	5,3	4,3	8,5	9,6
5	9,3	3,2	5,5	6,4	3,1	1,4
6	3,2	3,4	9,6	1,10	4,5	1,3

The objective is to solve the problem while minimizing the Makespan. In addition to Makespan there are a number of other performance measures but Makespan has been widely used as being simple and easy to implement.

The problem presented in Table 5.27 is loaded into a computer code based on the above described methodology (Figure 6.1) and developed in AM (Applications Manager, 2001). On completion of first iteration (first generation) the optimum result is obtained and is shown in Table 6.17, whereas, the Gantt chart representation of the best solution is given in Figure 6.12.

Table 6.17: Best Chromosome (Tariq et al. [2007])

1	**6**	**2**	**1**	**3**	**3**
2	**1**	**5**	**4**	**6**	**3**
4	**5**	**3**	**6**	**2**	**1**
3	**3**	**4**	**2**	**6**	**2**
2	**5**	**4**	**5**	**5**	**5**
1	**6**	**4**	**1**	**4**	**6**

Makespan = 55 time units.

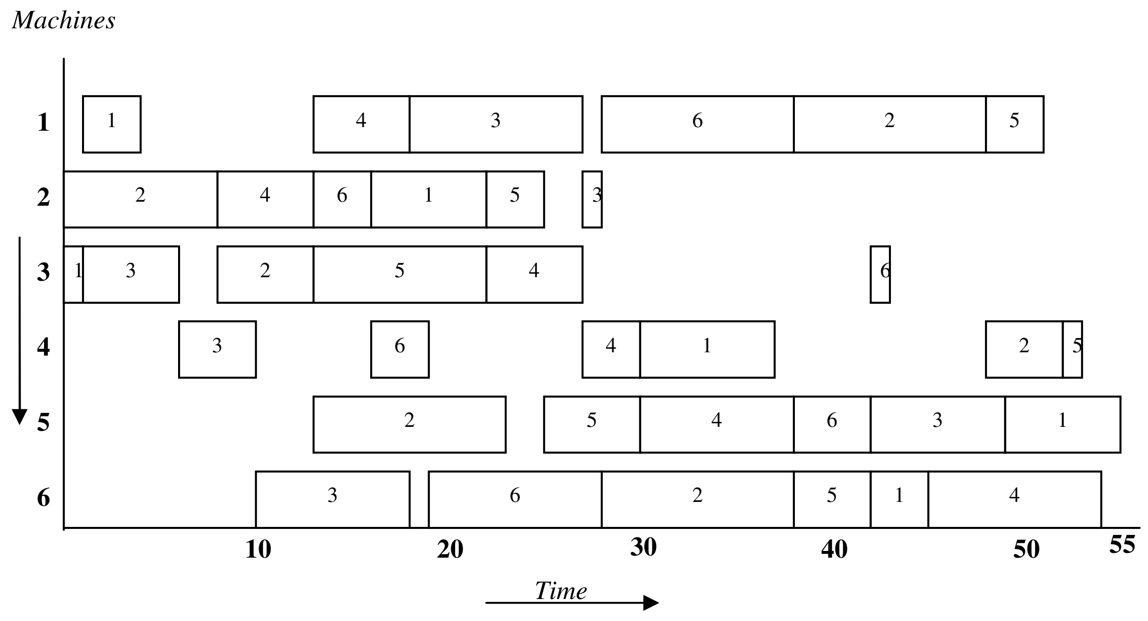


Figure 6.12: Schedule developed after decoding the best *Chrom.* (Tariq et al. [2007])

6.7 Sensitivity Analysis:

In order to carryout an analysis to determine the sensitivity of the hybrid GA developed for JSSP during this research a similar approach, described in Section 5.7, is adopted. For the purpose of this analysis a benchmark problem from Fisher & Thompson [1963] of order 10×10 is elected and solved for various: sizes of population, number of generations, mutation and crossover rates. Those values, of all these parameters, that resulted in minimum solution gape percentage are chosen to be used for all the other test problems. and finally those values of the parameters are selected at which the algorithm achieves the minimum percentage solution gap for the problem. The sensitivity of the algorithm against variations in the number of generations, population size, rate of crossover and mutation is presented in Figures 6.13, 6.14, 6.15 and 6.16 present the variation in percentage solution gape against variations of different GA parameters.

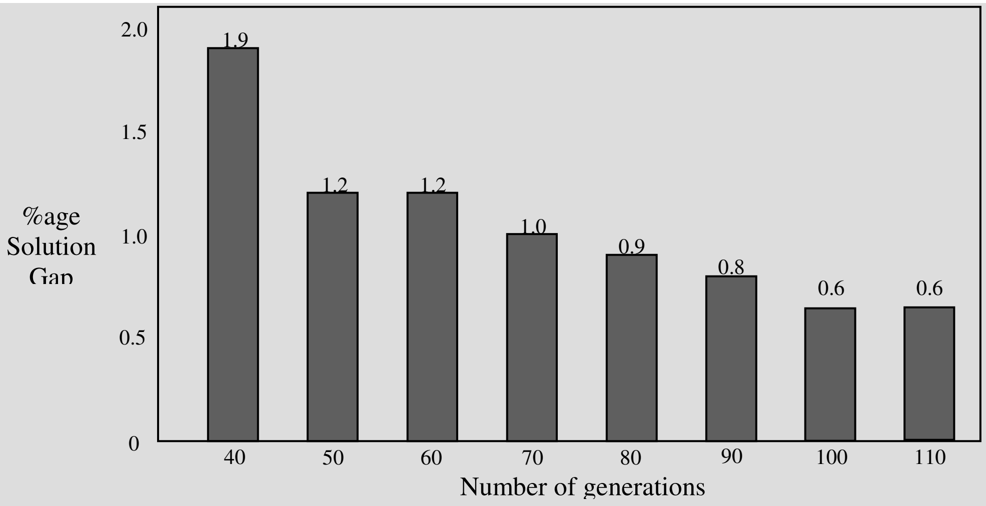


Figure 6.13: Effect of the number of generations on %age Solution Gap

Figure 6.13 shows that in a total number of 100 generations the minimum percentage solution gap is obtained. Further, as far as analysis regarding the population size, crossover and mutation rates are concerned that is presented in Figures 6.14, 6.15, 6.16 respectively. The analysis, on the whole, shows that a minimum value for the percentage solution gape is achieved for generations = 100, Population size = 75, crossover rate = 0.6 and mutation rate = 0.1.

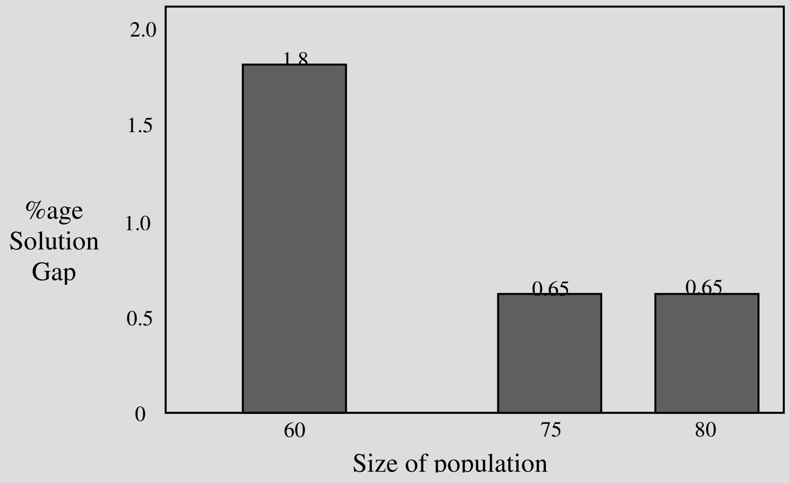


Figure 6.14: Effect of population size on %age Solution Gap

Number of generations = 100, Crossover rate = 0.6, Mutation rate = 0.1

Figures 6.15 and 6.16 present an analysis regarding crossover and mutation rates while using population size as 75 and a total number of generations as 100.

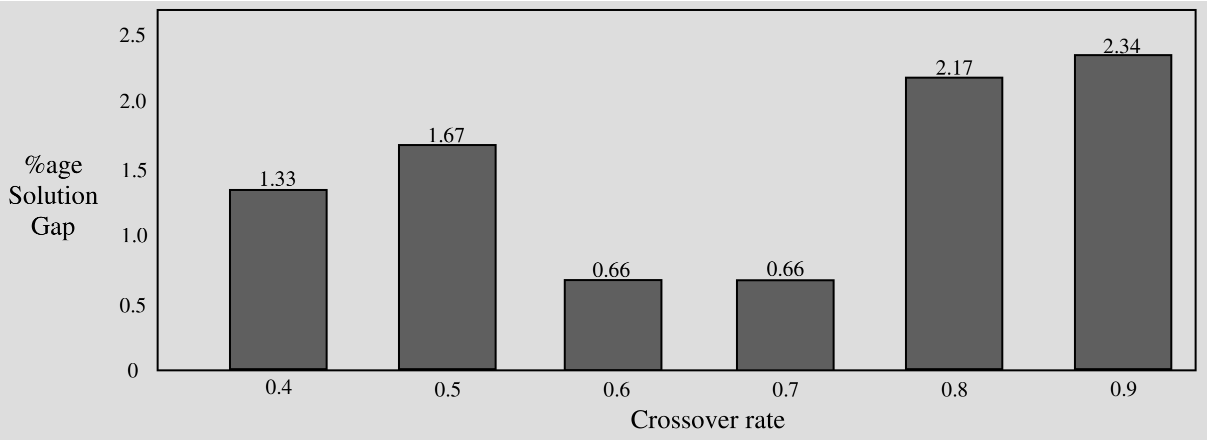


Figure 6.15: Effect of crossover rate on %age Solution Gap

Number of generations = 100, Population size = 75, Mutation rate = 0.1

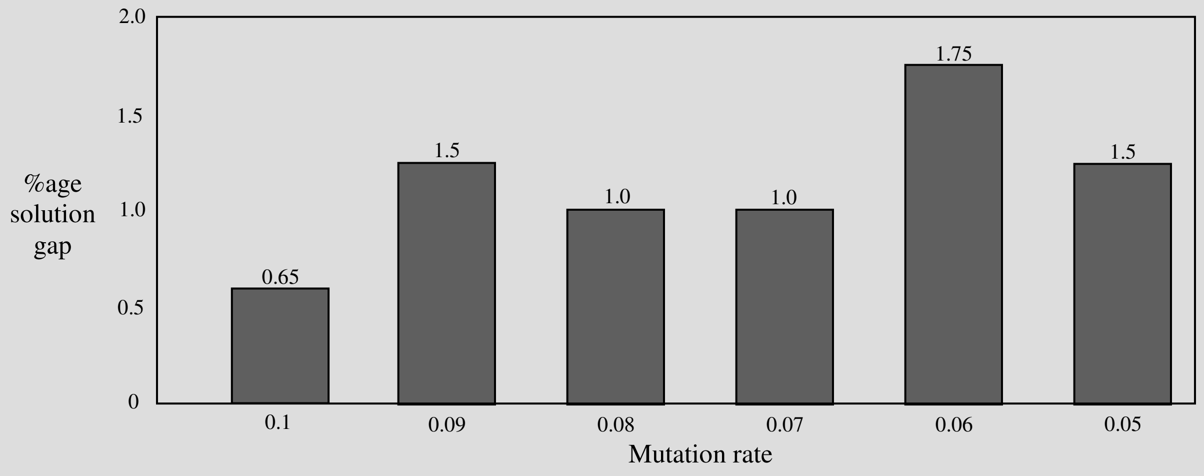


Figure 6.16: Effect of mutation rate on %age Solution Gap

Number of generations = 100, Population size = 75, Crossover rate = 0.6

It can be concluded from the analysis, about different GA parameters, presented above that for the satisfactory performance of the proposed algorithm the population size should be 75, the number of generation should be kept at 100, the crossover and mutation rates should be maintained at 0.6 and 0.1 respectively.

6.8 Combined Methodology for Operational Design of a CMS:

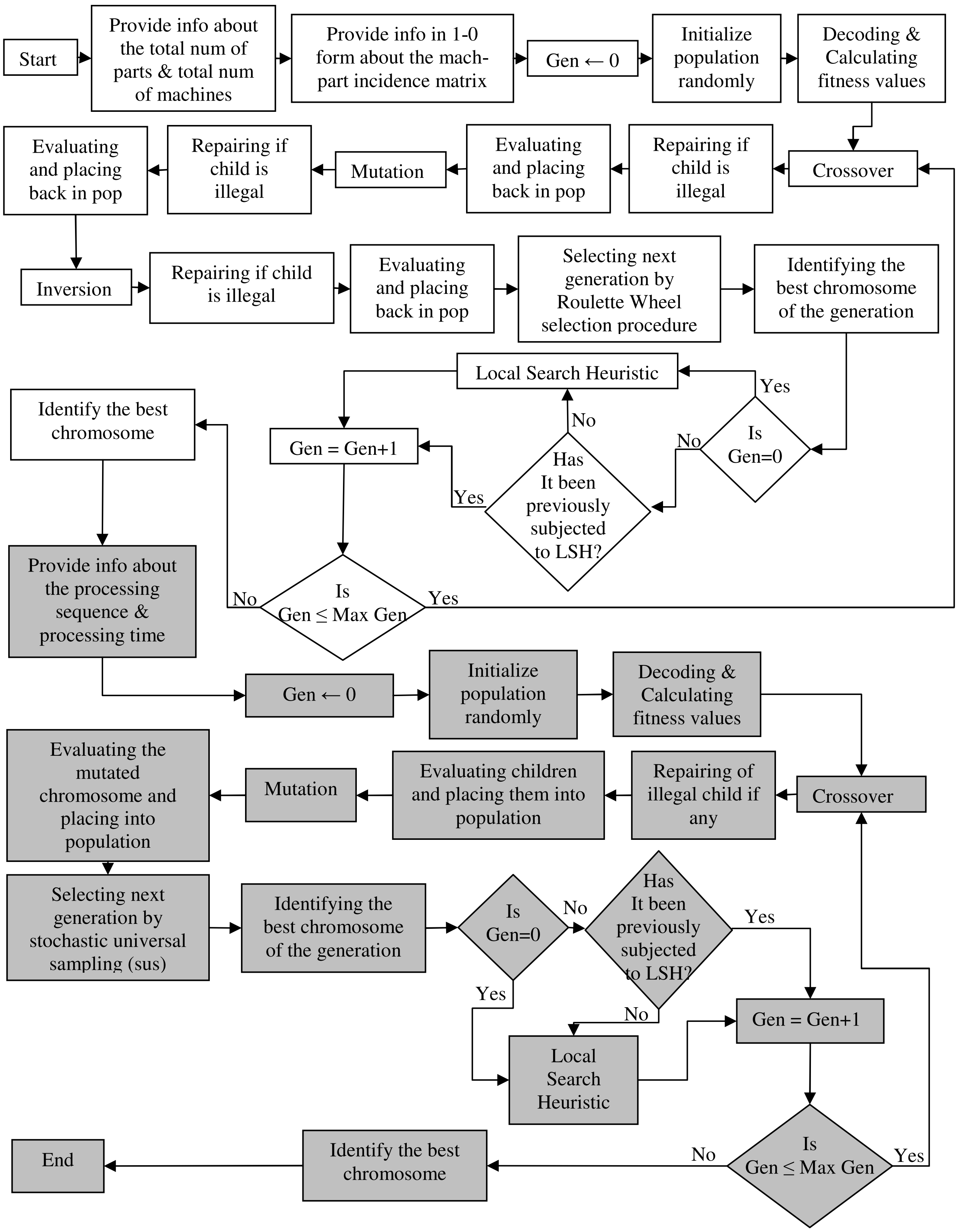
According to Wemmerlov and Hyer [1987] while designing a CMS several structural and operational issues must be taken into consideration. The first and most important task in the implementation of CM is the cell formation - grouping of parts into families and corresponding machines into cells. Wemmerlov and Hyer [1987] also pointed out that once the Machine-Part grouping has been carried out then the next step must be scheduling of the system and it is the job of the manufacturing engineer or operations manager to address the issue of allocating operations on each machine. This shows that carrying out an operational design of a CMS is more important rather than simply providing solution for the cell formation problem, as this would make an approach more practically useful.

As described above, operational design of a CMS consists of the following two steps:

1. CMS Design (Machine-Part grouping)
2. Scheduling of the system in a similar manner as Job-Shop scheduling.

Therefore, to carry out the operational design of CMS, the above two steps were followed and two separate tools have been developed: one for the Machine-Part grouping and another for scheduling of the system. The effectiveness of both the tools has been separately tested, verified and validated through a number of bench mark problems from literature. Since both the approaches have been found working satisfactorily, therefore this is the stage where they are Combined so that they can operate in a sequence.

In Figure 6.17 a block diagram representation for the operational design of a CMS is shown. A close look at the figure reveals that it is a combination of the two methodologies (Machine-Part grouping and scheduling of JSSP) described above. Figure 6.17 also shows that the output of the Machine-Part grouping tool is provided as an input to the Job-Shop scheduling tool.



Start
Provide info about the total num of parts & total num of machines
Provide info in 1-0 form about the mach-part incidence matrix
Gen ← 0
Initialize population randomly
Decoding & Calculating fitness values
Crossover
Repairing if child is illegal
Evaluating and placing back in pop
Mutation
Repairing if child is illegal
Evaluating and placing back in pop
Inversion
Repairing if child is illegal
Evaluating and placing back in pop
Selecting next generation by Roulette Wheel selection procedure
Identifying the best chromosome of the generation
Is Gen=0
Yes
No
Local Search Heuristic
Has It been previously subjected to LSH?
No
Yes
Gen = Gen+1
Is Gen ≤ Max Gen
No
Yes
Identify the best chromosome
Provide info about the processing sequence & processing time
Gen ← 0
Initialize population randomly
Decoding & Calculating fitness values
Crossover
Repairing of illegal child if any
Evaluating children and placing them into population
Mutation
Evaluating the mutated chromosome and placing into population
Selecting next generation by stochastic universal sampling (sus)
Identifying the best chromosome of the generation
Is Gen=0
No
Yes
Has It been previously subjected to LSH?
Yes
No
Local Search Heuristic
Gen = Gen+1
Is Gen ≤ Max Gen
Yes
No
Identify the best chromosome
End

Figure 6.17: Combined Methodology for operational design of a CMS

There are two main outputs of the program. First output is obtained from the Machine-Part grouping portion (the upper half of Figure 6.17) and that is the final Machine-Part incidence matrix which is in block-diagonalized form. The second output is given by the scheduling part (the lower half of Figure 6.17) and is in the form of a complete production schedule providing details about each operation of a job i.e. at what time and on which machine it is going to be performed.

6.9 Numerical Example of Combined Model

To further elaborate the combined working of the two methodologies, described previously in this chapter, let us select a grouping problem from literature (King and Nakornchai [1982]) and solve it with the help of the combined methodology being proposed here (Figure 6.17). The initial Machine-Part incidence for the problem is as shown in Table 6.18.

Table 6.18: Machine-Part incidence matrix (King and Nakornchai [1982])

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

The above initial Machine-Part incidence matrix is provided as an input along with the information about the total number of cells (two) that the system is to be divided into, to the Machine-Part grouping tool (hybrid GA based approach for Machine-Part grouping). The best result was obtained in first generation. The chromosome representing the best solution is as shown below:

Best [*i*] = [2 1 1 2 1   1 2 1 2 2 2 1]

The decoded form of the above result is a block-diagonalized form of the initial Machine-Part incidence matrix, as shown in Table 6.19.

Table 6.19: Final Machine-Part matrix

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

Using (5.4) (5.5) (5.7) and (5.2), respectively, the following can be calculated.

The total number of 1's in the Machine-Part incidence matrix (N_1) = 16

Total number of 0's inside the block diagonal (N_0^{in}) = 3

Total number of 1's outside the block diagonal (N_1^{out}) = 2

$$\text{GE} = \frac{N_1 - N_1^{out}}{N_1 + N_0^{in}} = \frac{16-2}{16+3} = 0.7368 = 73.68\%$$

Once the Machine-Part grouping phase is completed then information about the processing sequence and processing time is to be provided by the user. Here, for convenience, the processing sequence for each job and the processing time for each operation are generated randomly, as shown in Table 6.20 and Table 6.21, respectively.

Table 6.20: Processing sequence for each job

Jobs	Operations				
	1	**2**	**3**	**4**	**5**
1	1	5	2	3	4
2	2	4	1	5	3
3	4	2	1	5	3
4	4	2	3	1	5
5	1	2	3	5	4
6	4	2	3	1	5
7	4	3	1	2	5

Table 6.21: Processing time for each operation

Jobs	Operations				
	1	**2**	**3**	**4**	**5**
1	0	19	11	19	0
2	0	13	13	0	0
3	0	19	0	0	11
4	19	0	0	16	0
5	10	0	0	18	0
6	18	0	12	11	0
7	0	18	0	0	18

Processing time is randomly generated from an interval of (10,20)

Though all the values in Table 6.20 and Table 6.21 are randomly generated, but being displayed on screen (in program) in the form of input fields, the user would be having an opportunity to change these values as per the processing requirements of jobs in the system. Another point that one can argue about is that according to Table 6.20 there are certain operations for each job on certain machines which are mentioned in the sequence but according to the Machine-Part incidence matrix, jobs do not even have those operations for example: as per Table 6.20 Job 1 has its 1st and 5th operation on Machine 1 and Machine 4, whereas according to the Machine-Part incidence matrix (Table 6.18) Job 1 does not even have any operation on these Machines (1 and 4). The reason for such anomaly is that the hybrid GA based tool for scheduling of the system is in fact developed for the Job-Shop system, where each job has an operation on each machine. Therefore while providing information about the operation sequence of each job, those machines must also be included in the sequence on which that job does not even have any operation. However, the computer code, initially developed for the scheduling of JSSP, has been modified to exclude such operations by assigning a processing time of zero to all such operations and processing time for all the other operations are randomly generated between 10 and 20, as shown in Table 6.21.

The information about total number of jobs, total number of machines, processing sequence, and corresponding processing times is provided as an input to the scheduling tool (hybrid GA for JSSP), described in Section 5.11. In order to save computational effort a Lower Bound (LB) value for the Makespan is calculated, as shown in Table 6.22 and after each generation the minimum Makespan value is compared with it. If at any point, before reaching the maximum number of generations, the LB value is reached then further search is stopped.

Table 6.22: Calculation of Lower Bound (LB)

Jobs	Machines					Sum of rows
	1	**2**	**3**	**4**	**5**	
1	0	11	19	0	19	49
2	13	0	0	13	0	26
3	0	19	11	0	0	30
4	16	0	0	19	0	35
5	10	0	0	0	18	28
6	11	0	12	18	0	41
7	0	0	18	0	18	36
Sum of columns	50	30	**60**	50	55	

Lower Bound (LB) = 60

Although in certain problems, the LB cannot be achieved because of complex processing sequence, but still it is an ideal target that keeps the search for best Makespan going and in certain cases compels the algorithm to utilize the maximum number of generations.

Now, allowing the program to run through the maximum number of generations (100) while keeping in check the LB value, the best solution obtained on termination is presented in Table 6.23.

Table 6.23: Chromosome/solution after 1st generation

7	1	7	2	3
6	6	4	7	1
6	5	4	4	4
3	3	6	2	3
4	6	1	2	7
2	5	5	5	3
7	1	5	2	1

Makespan = 70 time units

Using the decoding procedure presented in Figure 6.3, the following Gantt chart representation (Figure 6.18), for the schedule encoded in the chromosome shown in Table 6.23, can be developed.

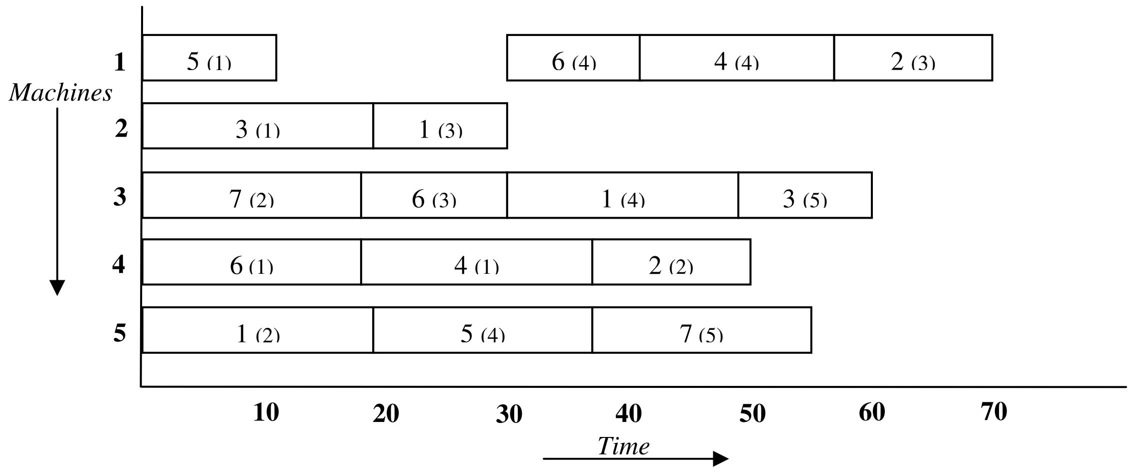


Figure 6.18: Schedule developed after decoding the best *Chrom*.

Although the LB value is 60, the Makespan displayed in Figure 5.34 is 70. This does not mean that the result cannot be optimum. Figure 6.18 shows that except for Machine 1, the waiting time for all the other machines is zero. Machine 1 has a total waiting time of 20 time units, because of the intercellular move of part 6. Part 6 has its 2nd operation on Machine 3, for which it has to travel from Cell 2 to Cell 1 and back to Cell 1 for its third operation.

The distinguishing feature of the above combined solution approach is that not only a grouping solution is provided like most of the techniques available in literature, but a solution for the scheduling part of the problem is also provided which definitely makes this technique more useful in practice.

6.10 Summary:

Since the problem of operational design of a CMS can be broadly categorised as the Machine-Part grouping problem and the cell scheduling problem, therefore the solution methodology, developed during this research, has also been divided into two main portions. In the first portion (Chapter 5) the Machine-Part grouping problem is solved, whereas in second portion (This Chapter) solution for the cell scheduling problem is provided. Separate tools have been developed for both the portions and their effectiveness have also been validated through a number of benchmark problems from literature (details available in the Next Chapter).

This Chapter has specifically provided a detailed description of the tool developed for solving the cell scheduling problem, based on the principles of JSSP, and then its combination with the tool for Machine-Part grouping which has been described in detail in Chapter 5. A similar solution approach, as in the case of Machine-Part grouping, has been adopted for solving the JSSP. Here also, GA is combined with an LSH using multipoint crossover, swap type mutation, and Stochastic Universal Sampling (SUS) as the selection procedure. The best solution in each generation is subjected to LSH provided that it has not been previously subjected to it. For better understanding of the programming logic each step of the algorithm is clearly explained both with the help of flow diagram and stepwise procedure. The methodology is further elaborated by solving a benchmark problem with the help of the proposed algorithm. To justify the values of the GA parameters (crossover = 60%, mutation = 10%, population size = 75, number of generations = 100) a sensitivity analysis is also presented.

After separate development, testing and validation of both the tools, they have been finally combined with each other. The combination of two approaches is carried out in such a manner that the output of Machine-Part grouping is used as an input by the cell scheduling part. The distinguishing feature of this combination is that it not only provides solution for the Machine-Part grouping problem but also solves the cell scheduling problem. It makes this approach more useful in practice as compared to all those design approaches which consider the CMS design problem as only a Machine-Part grouping problem.

CHAPTER 7

RESULTS AND DISCUSSIONS

7.1 Introduction:

This chapter presents testing and validation of the two models (hybrid GA for Machine-Part grouping and hybrid GA for JSSP), developed during this research and described in Chapter 5 and 6. Regarding testing and validation of the models, details of the selected benchmark problems, their solutions, analysis of results and comparison with other techniques are described in detail in this chapter. This chapter also provides an insight, as far as the combined working of the two tools is concerned, in the form of an analysis based on results of some problems which are solved by the combined approach i.e. first Machine-Part grouping and then cell scheduling.

7.2 Performance and Analysis of Hybrid GA for Machine-Part Grouping:

To check the performance of the hybrid model developed during this research for Machine-Part grouping, a set of 36 different benchmark problems have been selected from literature. The selected problems are of variable sizes as far as their Machine-Part incidence matrices are concerned. This is done deliberately so that performance of the proposed approach can be tested on different data sets.

The matrix size of each problem and its source is mentioned in Table 7.1.

Table 7.1: Selected problems from literature (Tariq et al. [2006] & [2009])

S/No.	Source	*Machs×Jobs*	Matrix size
1	King and Nakornchai [1982]	5×7	35
2	Waghodekar and Sahu [1984]	5×7	35
3	Seifoddini [1989]	5×18	90
4	Kusiak [1992]	6×8	48
5	Kusiak and Chow [1987]	7×11	77
6	Boctor [1991]	7×11	77
7	Seiffodini and Wolfe [1986]	8×12	96
8	Chandrasekharan and Rajagopalan [1986a]	8×20	160
9	Chandrasekharan and Rajagopalan [1986b]	8×20	160
10	Fernando and Mauricio [2002]	9×11	99
11	Mosier and Taube [1985a]	10×10	100
12	Chan and Milner [1982]	10×15	150
13	Askin and Subrammanian [1987]	14×23	322
14	Stanfel [1985]	14×24	336

15	McCormick et al. [1972]	16×24	384
16	Srinivasan et al. [1990]	16×30	480
17	King [1980]	16×43	688
18	Carrie [1973]	18×24	432
19	Mosier & Taube [1985b]	20×20	400
20	Kumar et al. [1986]	20×23	460
21	Carrie [1973]	20×35	700
22	Boe and Cheng [1991]	20×35	700
23	Chandrasekharan & Rajagopalan [1989]-1	24×40	960
24	Chandrasekharan & Rajagopalan [1989]-2	24×40	960
25	Chandrasekharan & Rajagopalan [1989]-3	24×40	960
26	Chandrasekharan & Rajagopalan [1989]-5	24×40	960
27	Chandrasekharan & Rajagopalan [1989]-6	24×40	960
28	Chandrasekharan & Rajagopalan [1989]-7	24×40	960
29	McCormick et al. [1972]	27×27	729
30	Carrie [1973]	28×46	1,288
31	Kumar and Vannelli [1987]	30×41	1,230
32	Stanfel [1985]	30×50	1,500
33	Stanfel [1985]	30×50	1,500
34	King & Nakornchai [1982]	30×90	2,700
35	McCormick et al. [1972]	37×53	1,961
36	Chandrasekharan & Rajagopalan [1987]	40×100	4,000

Table 7.1 can also be interpreted in the form of a graphical representation as shown in Figure 7.1. It clearly shows the variation from a minimum value of 35 (product of total number of machines and total number of parts) to a maximum value of 4000, hence covering a reasonably wide range of different problem sizes and providing enough challenge to the algorithm to prove its effectiveness.

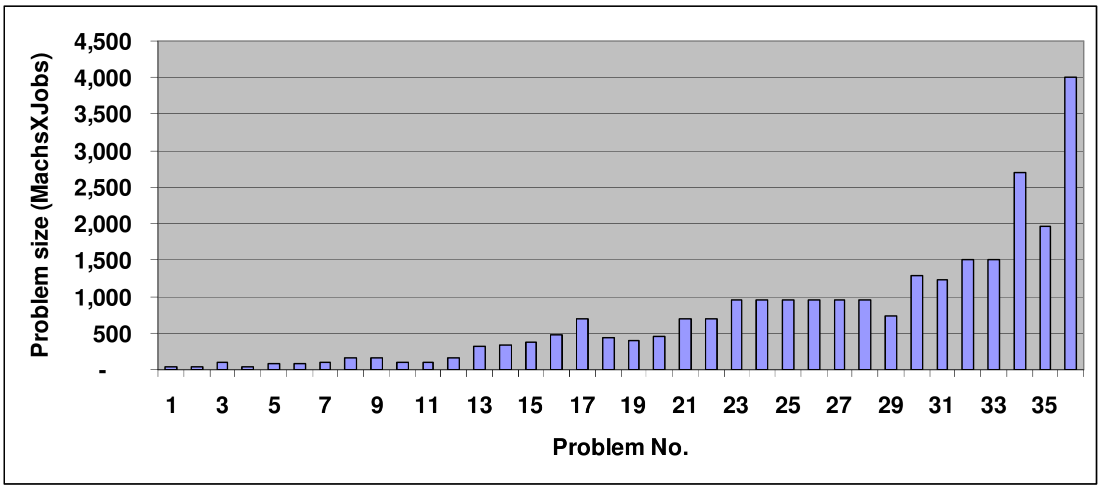


Figure 7.1: Variation in size of Machine-Part incidence matrix

After solving all the problems listed in Table 7.1, the GE reported for the same problems by the following techniques is compared with the GE obtained during this research. (Tariq et al. [2006] & [2009])

a) ZODIAC (Chandrasekharan and Rajagopalan, [1987])
b) GRAFICS (Srinivasan and Narenderan, [1991])
c) CA - Clustering algorithm (Srinivasan, [1994])
d) GATSP - Genetic algorithms (Cheng et al. [1998])
e) GA - Genetic algorithm (Onwubolu & Mutingi, [2001])
f) GP- Genetic programming (Dimopoulos & Mort, [2001])
g) Hyb. GA - Hybrid GA (Fernando and Mauricio, [2002])

Best results for the selected benchmark problems from literature, as far as the GE values are concerned, have been reported by the above listed techniques. Therefore in order to verify the effectiveness of the approach developed during this research a comparison, based on the selected benchmark problems, is carried out with the above listed techniques and is presented in Table 7.2. Table 7.2 also displays the CPU time for each problem. This time is recorded while solving these problems on a machine having 1.86 GHz Intel® T2130 processor with a RAM of 1.0 GB.

Table 7.2: Test results (GE values) of the selected problems from literature (Tariq et al. [2006] & [2009])

No.	ZODIAC [1987]	GRAFICS [1991]	CA [1994]	GA TSP [1998]	GP [2001]	GA [2001]	Hyb. GA [2002]	Gens	This Approach	Gens	CPU Time (Seconds)
1	73.68	73.68	-	-	-	-	73.68	1	73.68	1	0.42
2	56.52	60.87	-	68.00	-	62.50	62.50	1	69.57	1	0.51
3	77.36	-	-	77.36	-	77.36	79.59	1	79.59	1	0.67
4	76.92	-	-	76.92	-	76.92	76.92	1	76.92	1	0.56
5	39.13	53.12	-	46.88	-	50.00	53.13	6	58.62	1	2.60
6	70.37	-	-	70.37	-	70.37	70.37	1	70.37	1	0.89
7	68.30	68.30	-	-	-	-	68.30	1	68.30	1	1.40
8	58.33	58.13	58.72	58.33	58.72	55.91	58.72	2	58.72	1	18.72
9	85.24	85.24	85.24	85.24	85.24	85.25	85.25	1	85.25	1	14.12
10	-	-	-	-	-	-	86.67	1	86.67	1	0.52
11	70.59	70.59	70.59	70.59	-	-	70.59	1	70.59	1	1.52
12	92.00	92.00	-	92.00	-	-	92.00	1	92.00	1	2.12
13	64.36	64.36	64.36	-	-	-	69.86	10	70.83	1	22.64

14	5	65.55	-	67.44	-	63.48	69.33	1	70.51	1	3.23
15	32.09	45.52	48.70	-	-	-	51.96	21	51.96	2	9.82
16	67.83	67.83	67.83	-	-	-	67.83	1	67.83	1	4.02
17	53.76	54.39	54.44	53.89	-	-	54.86	1	54.86	1	11.06
18	41.84	48.91	44.20	-	-	-	54.46	32	54.95	2	9.89
19	21.63	38.26	-	37.12	-	34.16	42.96	50	43.45	8	21.48
20	38.66	49.36	43.01	46.62	49.00	39.02	49.65	78	49.65	1	4.63
21	75.14	75.14	75.14	75.28	-	66.30	76.14	1	76.14	1	5.06
22	51.13	-	-	55.14	-	44.44	58.07	2	58.38	1	10.29
23	100.00	100.00	100.00	100.00	100.00	100.00	100.00	1	100.00	1	15.29
24	85.11	85.11	85.11	85.11	85.11	-	85.11	1	85.11	1	18.47
25	73.51	73.51	73.51	73.03	73.51	73.03	73.51	1	73.51	1	23.75
26	20.42	43.27	51.81	49.37	-	37.62	51.85	114	52.50	1	49.11
27	18.23	44.51	44.72	44.67	-	34.76	46.50	117	46.84	12	231.09
28	17.61	41.67	44.17	42.50	-	34.06	44.85	75	44.85	8	143.91
29	52.14	41.37	51.00	-	-	-	54.27	8	54.31	4	21.07
30	33.01	32.86	40.00	-	-	-	43.85	117	46.43	7	204.83
31	33.46	55.43	55.29	53.80	-	40.96	57.69	111	60.74	4	197.62
32	46.06	56.32	58.70	56.61	-	48.28	59.43	113	59.66	35	1626.3
33	21.11	47.96	46.30	45.93	-	37.55	50.51	93	50.51	35	1391.12
34	32.73	39.41	40.05	-	-	-	41.71	45	44.67	9	823.52
35	52.21	52.21	-	-	-	-	56.14	1	59.60	8	55.37
36	83.66	83.92	83.92	84.03	84.03	83.90	84.03	3	84.03	1	270.76

(All the rows having a grey coloured background, show that the results have been further improved)

Analysing the results presented in Table 7.2, it can be clearly seen that the approach developed during this research has given GE for all the benchmark problems which is either greater than or equal to the previously reported values and that is why it would not be an overstatement to say that this research has outperformed all the previous research approaches in terms of accuracy and consistency. By accuracy we mean the value of GE which for none of the benchmark problems is less than the previously reported values, whereas in 15 problems (41.67% of the total tested problems) it is even better than the previously reported results. On the other hand by consistency we mean that this behaviour of giving accurate results remains consistent throughout i.e. right from the first problem till the 36th problem, whereas all the other techniques, used here for comparison, loose their accuracy as the problem size increases. It is evident from Table 7.2 that out of the first 18 results (where the Machine-Part incidence matrix size varies from 35 to 688 entries) only 5 (27.77%)

have been further improved whereas in the last 18 results (where the Machine-Part incidence matrix size varies from 400 to 4000 entries), 10 (55.55%) results have been further improved.

A graphical representation of results presented in Table 7.2 is shown in Figure 7.2.

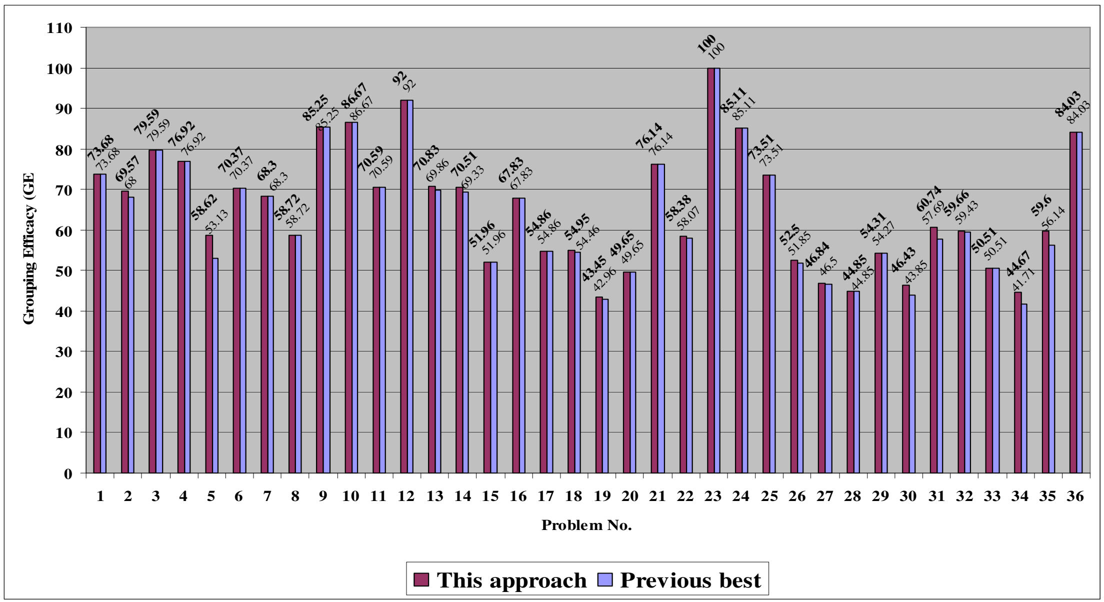


Figure 7.2: Comparison of GE obtained by this research and the best reported in literature

It can be seen, both, in Table 7.2 and Figure 7.2 that the approach, developed during this research, have produced results which are either better or equivalent to the previously best reported results in literature. For 58.33% (21 problems) of the total tested problems (36 problems) the results are equivalent to the best reported in literature. For 41.67% (15 problems) the GE values obtained by this approach are higher than the previously reported values. Another distinguishing feature of the approach is that in case of 66.67% (24 problems) it took only one generation to reach the best value of GE which proves that the LSH developed during this research is very effective and it has made this approach comparatively more accurate and consistent, whereas the all the other approaches used here for comparison have the tendency of loosing accuracy as the size of the problem increases and the incidence matrices become more ill-structured. In Table 7.2 it can be seen that in 31.82% (7 problems) of the first 22 results, where the machine part incidence matrix size is less than or equal to 700 entries, the value of GE is higher than the previous best reported. To further establish this claim let us consider the results of the last 14 problems i.e. from problems 23 to 36 in which the matrix size is either greater than or equal to 960 entries (24×40), it can be very clearly observed that 57.14% (8 results) are better than previous bests. To dig it even further we can see that in 75 % (6 results) of the last 8 results, where the Machine-Part incidence matrix size is either greater than or equal to 729 entries (27×27), have displayed higher GE values. These statistics prove that the technique developed during this research is comparatively more consistent and is not much affected by the size and structure of the Machine-Part incidence matrix (Tariq et al. [2006] & [2009]).

The above improvements are because of the strategy of simultaneous Machine-Part grouping, used during this research, and the development and placement of a strong LSH at the heart of the traditional GA loop. The LSH, developed during this research, is organised in such a manner that it changes the position of a part and/or machine from one cell to another while checking any consequent increase in the value of GE. The procedure is started from the first gene and by the time it reaches the last gene a comparatively better solution, having higher GE, has evolved. The LSH developed during this research, though does not take all the available options (placing a part and/or machine in all the available cells, one by one) into consideration, considers enough number of options that makes it comparatively more effective. In comparison, the technique developed by Fernando & Mauricio [2002] does not group machines into cells and parts into families simultaneously and at the same time LSH is first used to make part families in accordance with the initial machine groups and then refines the machine arrangement into a final arrangement on the available part families. This shows that their technique is sequential in nature rather than simultaneous and therefore the results produced, in comparison to this research, are less accurate. Also, since the LSH, developed during this research, is applied both to the formation of machine groups and corresponding part families and considers larger number of options for improvement (search is not limited in the neighbourhood only) therefore it is more rigorous as compared to the one presented by Fernando & Mauricio [2002].

A graphical representation, of the analysis being carried out in the preceding paragraphs, is shown in Figure 7.3, where all the problems are divided into 6 groups each containing 6 problems.

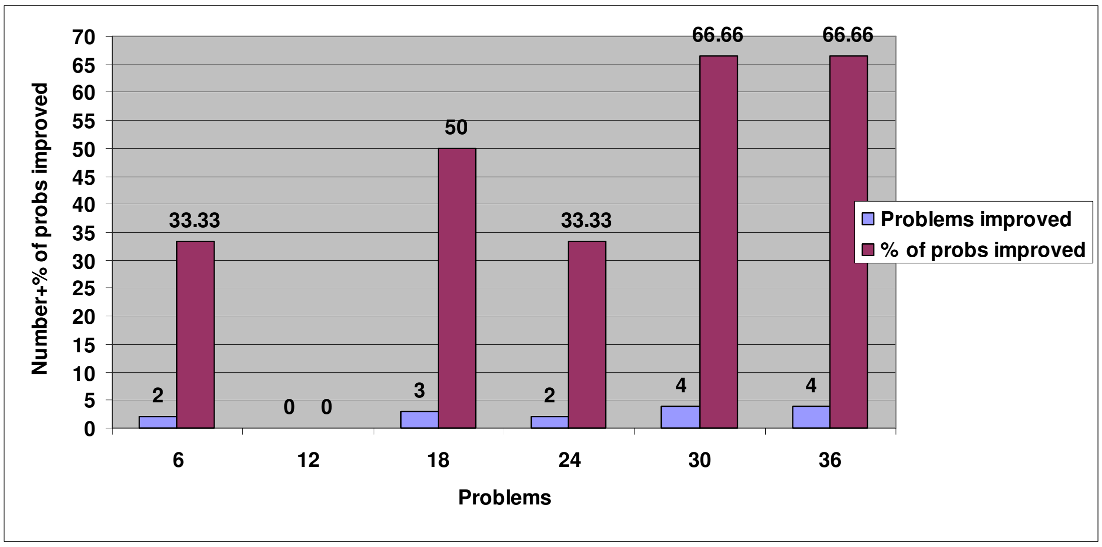


Figure 7.3: The number and percentage of problems improved

It can be clearly seen in Figure 6.3 that the maximum number of problems (66.66%) are being further improved in the groups from 25 to 30 and from 31 to 36, where the problem size varies from 960 to 4000 entries. This shows the consistency of the algorithm in terms of accuracy.

Another comparison of this approach, on the basis of the total number of generations required by GA to reach a maximum value, is also made with the technique developed by Fernando and Mauricio [2002]. A graphical representation of this comparison is shown in Figure 7.4.

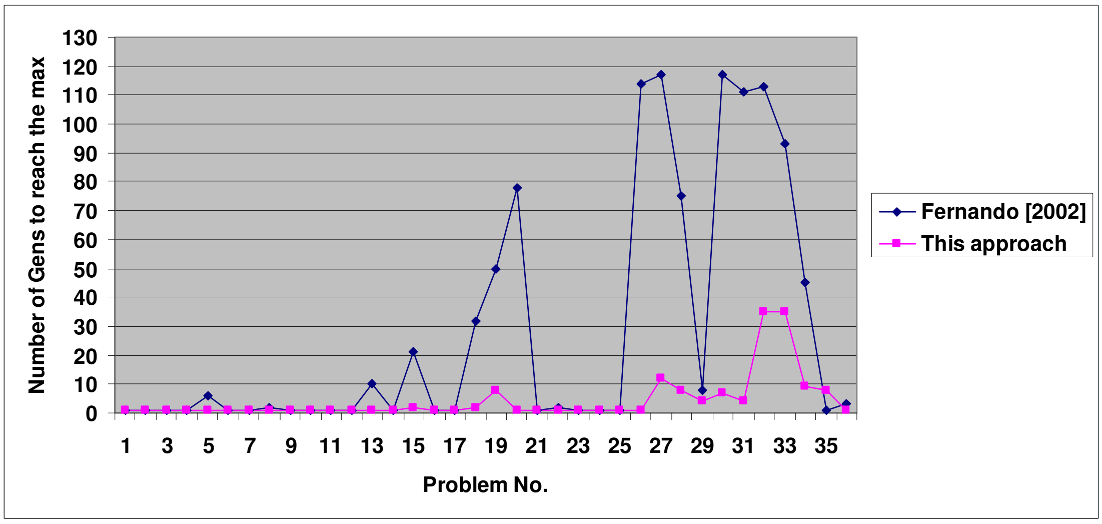


Figure 7.4: Comparison (between Fernando's work and this approach) in terms of the number of generations to reach the maximum value of GE for each problem

It is clear from Figure 7.4 and Table 7.2 that the approach developed during this research needed less number of generations in more than 50% of the problems to reach the maximum value of GE, as compared to Fernando and Mauricio [2002]. On the other hand, for all the rest of the problems it needed an equivalent number of generations except at one instant (Problem 35). This clearly proves the power and effectiveness of the LSH being proposed here. It has already been described that this effectiveness is due to the fact that it (The LSH) does not limit its search to immediate neighbourhood. It takes a large number of options into consideration while improving a solution which makes it comparatively more effective. Since the LSH is placed at the heart of the traditional GA loop and further improves the best solution of each generation, therefore it reaches the optimum/near optimum GE value in comparatively lesser number of generations.

In order to further elaborate the ability of this approach to produce comparatively better results, percentage improvement has been calculated for each of the tested problems and is presented in Table 7.3.

Table 7.3: Results in terms of percentage improvement

Problem #	Previous best (GE)	This approach (GE)	Percentage improvement
1	73.68	73.68	0
2	68.00	69.57	2.31
3	79.59	79.59	0
4	76.92	76.92	0
5	53.13	58.62	10.33
6	70.37	70.37	0
7	68.30	68.30	0
8	58.72	58.72	0
9	85.25	85.25	0
10	86.67	86.67	0
11	70.59	70.59	0
12	92.00	92.00	0
13	69.86	70.83	1.38
14	69.33	70.51	1.70
15	51.96	51.96	0
16	67.83	67.83	0
17	54.86	54.86	0
18	54.46	54.95	0.8997
19	42.96	43.45	1.145
20	49.65	49.65	0
21	76.14	76.14	0
22	58.07	58.38	0.5338
23	100	100	0

24	85.11	85.11	0
25	73.51	73.51	0
26	51.85	52.50	1.254
27	46.50	46.84	0.7312
28	44.85	44.85	0
29	54.27	54.31	0.074
30	43.85	46.43	5.884
31	57.69	60.74	5.287
32	59.43	59.66	0.387
33	50.51	50.51	0
34	41.71	44.67	7.097
35	56.14	59.60	6.163
36	84.03	84.03	0

The data presented in Table 7.3 can also be displayed in the form of a graphical representation as shown in Figure 7.5.

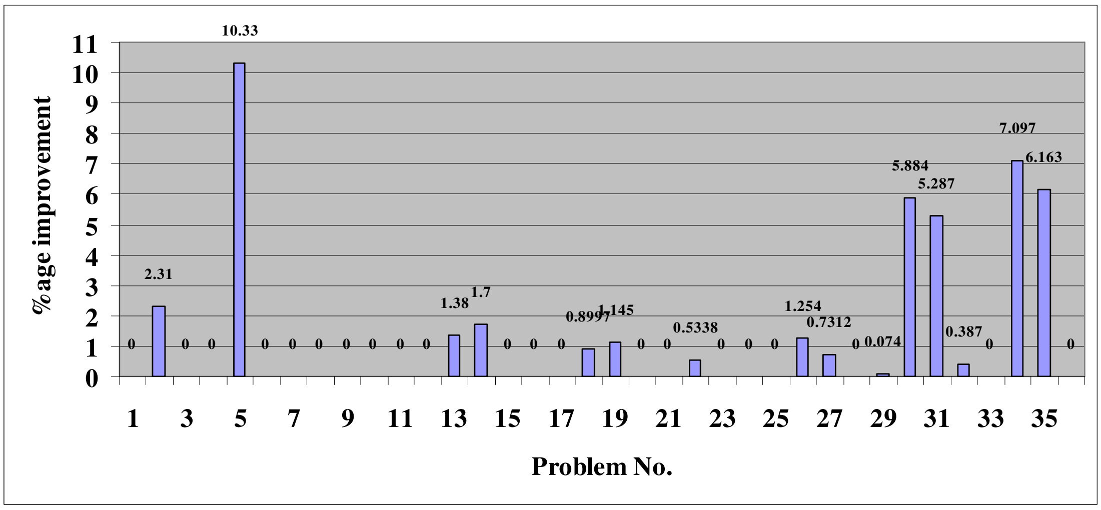


Figure 7.5: %age improvement in result (GE) of each problem

Table 7.3 and Figure 7.5 show that percentage improvement has been recorded in case of 15 (41.67%) problems. The maximum improvement (10.33%) has been recorded in problem 5, whereas the minimum (0.074%) has been recorded in problem 29. However, on the average an improvement of 1.255% has been recorded.

7.3 Statistical Analysis: (Tariq et al. [2009])

In order to validate the results, obtained during this research, a statistical analysis is presented. For this purpose the two techniques, New approach (this research) and Hybrid GA (Fernando & Mauricio [2002]), have been compared with each other using a 95% Confidence Interval (CI) for the mean difference while performing a Paired T-Test. It is to be noted here that zero is not included in the confidence interval between the two techniques for the mean difference. This indicates that the two techniques are different from each other. Inconsistency of the data is indicated by the small p-value (p = 0.006) with H0: μ d = 0, which suggests that the two approaches did not perform equally rather the performance of the new approach (mean = 65.3222) was comparatively better than the hybrid approach (mean = 64.5081) as far as calculating the value of GE is concerned for the total number of 36 problems selected form literature (Table 6.1 & Table 6.2).

Paired T-Test and CI: Hyb.GA (Fernando & Mauricio, 2002), New Approach (this research) (Tariq et al. [2009])

```
Paired T for Hyb.GA - New Approach

                 N        Mean     StDev   SE Mean
Hyb.GA          36     64.5081   15.1927    2.5321
New Approach    36     65.3222   14.7703    2.4617
Difference      36   -0.814167  1.660087  0.276681

95% CI for mean difference: (-1.375859, -0.252474)
T-Test of mean difference = 0 (vs not = 0): T-Value = -2.94: P-Value = 0.006
```

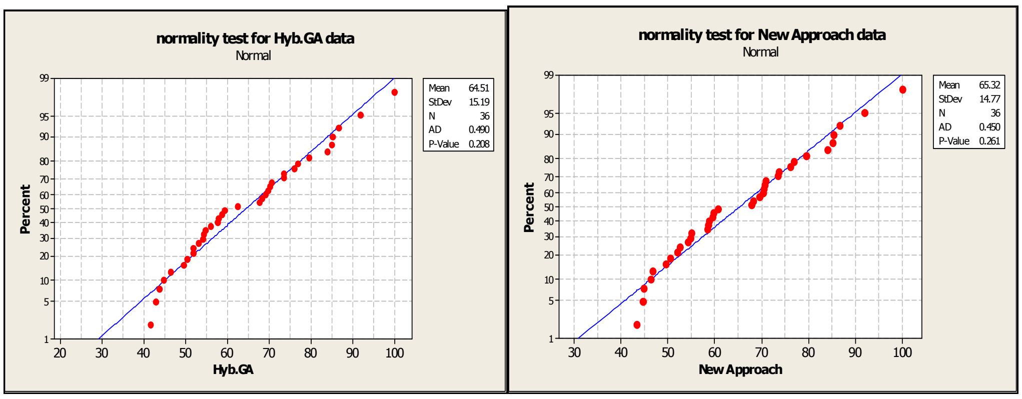


Figure 7.6: Normality test for Hybrid GA (Fernando & Mauricio [2002]) and New Approach (This research) (Tariq et al. [2009])

The plot of normal probabilities versus the data is the graphical output as shown in Figure 6.6. The departure of data from the fitted line in the extremes (distribution tails) can be more evidently viewed. But since the assumption of normality in any T-Test is of moderate importance only, therefore the P-value of the Anderson Darling (AD) test suggest that it is safe enough to apply the Paired T-Test, in spite of the fact that the data looks like departing in lower extremes from the fitted line.

Our contention is also supported by the graph obtained from the test of equal variances.

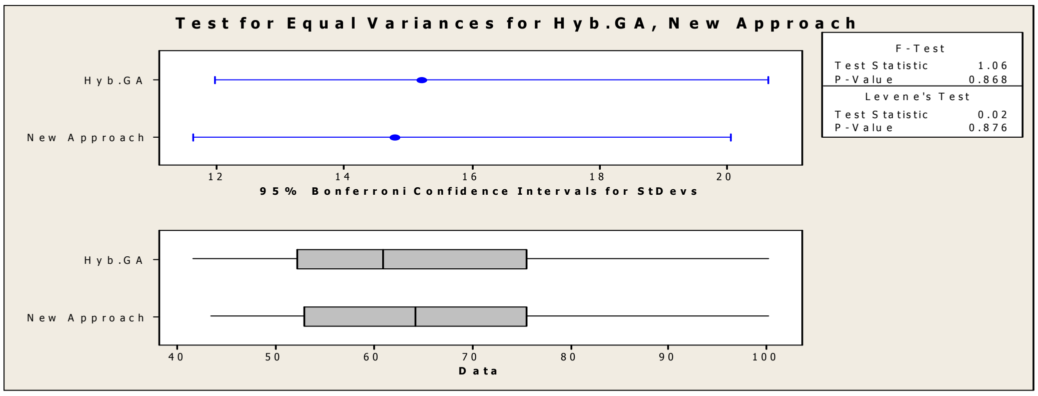


Figure 7.7: Equal Variance test for Hybrid GA and New Approach (Tariq et al. [2009])

7.4 Formation of Single Machine Cells: (Tariq et al. [2007])

Some of the research works, presented in literature, advocates restriction on the formation of single machine cells e.g. Fernando and Mauricio [2002] discarded any such solutions from population in which single machine cells were formed. Contrary to this, it has been found during this research that the formation of single machine cells is in fact beneficial in terms of increase in GE and decrease in intercellular moves which consequently reduces the material handling cost. In order to show the effectiveness of allowing the formation of single machine cells an analysis is presented in the following Section.

7.4.1 Numerical Example (Formation of Single Machine Cells): (Tariq et al. [2007])

To judge the effectiveness of allowing the formation of single machine cell, a numerical example is selected from Waghodekar and Sahu [1984]. The problem consists of a total number of 7 parts and 5 machines. The total number of cells to be developed is 2. Table 6.4 presents the initial Machine-Part incidence matrix of the problem. First solution for the problem is obtained while restricting the formation of cells that posses one machine. Afterwards the same problem is solved while allowing the formation of cells that posses one machine and finally the results are compared:

Table 7.4: The initial machine part incidence matrix (Tariq et al. [2007])

Machs. Parts	**1**	**2**	**3**	**4**	**5**
1	1	0	0	1	0
2	0	1	0	1	1
3	0	1	1	1	0
4	0	1	1	1	1
5	1	1	1	0	1
6	1	0	1	0	1
7	1	0	0	0	0

First the algorithm is modified to restrict the formation of single machine cell. The solution obtained is as shown in Table 7.5:

Table 7.5: Solution without single machine cell (Tariq et al. [2007])

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

Table 7.5 shows that a total number of five 1s are outside the block diagonal. This means that 5 parts (2, 3, 4, 5 and 6) require processing in other cells i.e. other than the one to which they are allocated. Since each of these parts would be carried to another cell for processing and brought back afterwards, therefore the total number of intercellular moves would be twice the total number of 1s outside the block diagonal. All the relevant calculations are as follows:

Grouping Efficacy (GE) = 62.5%

Number of 1s outside the block diagonal = 5

Total number of intercellular (IC) moves = 5 × 2 = 10

Let, material handling cost per IC move = 10 units.

So, total material handling cost for 10 IC moves = 10 × 10 = 100 units

Now, secondly, the algorithm is converted back to its original form i.e. formation of single machine cells is allowed. This time the solution, for the same example, obtained is as shown in Table 7.6:

Table 7.6: Solution with single machine cell (Tariq et al. [2007])

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

It can be clearly seen in Table 7.6, that this time the total number of 1s outside the block diagonal are 4 instead of 5. The total number of 0s inside the block diagonal has also decreased from 4 to 3. The net effect of this decrease in the total number 1s outside, and 0s inside the block diagonal is an increase in the value of GE and decrease in the material handling cost, as shown below:

Grouping efficacy (GE) = 69.57%

Number of 1s outside the block diagonal = 4

Total number of IC moves = $4 \times 2 = 8$

Total material handling cost for 8 IC moves = $8 \times 10 = 80$ units

Total material handling cost saved = $100 - 80 = 20$ units.

7.4.2 Computational Results (Formation of Single Machine Cells):

The phenomenon, of increase in GE and decrease in material handling cost as a result of allowing the formation of single machine cells is not limited only to the example presented above. The same trend is experienced in a number of other, already tested, benchmark problems as well. A total number of 8 such problems (out of the total 36 tested problems) were found in which the final result contained single machine cells. The matrix size and source of these problems are as shown in Table 7.7.

Table 7.7: Problems with their sizes and sources (Tariq et al. [2007])

Pb/No.	Source	Size
1	Stanfel (1985)	14×24
2	Carrie (1973)	18×24
3	Mosier & Taube (1985b)	20×20
4	Boe and Cheng (1991)	20×35
5	Chandrasekharan & Rajagopalan (1989)-5	24×40

6	Carrie (1973)	28×46
7	Kumar and Vannelli (1987)	30×41
8	Stanfel (1985)	30×50

Results of the problems listed in Table 7.7 are as shown in Table 7.8, in the following:

Table 7.8: Results of the problems shown in Table 7.7 (Tariq et al. [2007])

Pb.	Prev. Best (GE) without single mach cells	Prev. Best (IC moves) without single mach cells	This Paper (GE) with single mach cells	This Paper (IC moves) with single mach cells	Diff. in IC moves (d)	Total IC moves saved (2×d)	Cost Saved d×10 Units
1	69.33	09	70.51	06	03	06	60
2	54.46	27	54.95	27	00	00	00
3	42.96	53	43.45	48	05	10	100
4	58.07	41	58.38	41	00	00	00
5	51.85	47	52.50	47	00	00	00
6	43.85	97	46.43	94	03	06	60
7	57.69	38	60.74	29	09	18	180
8	59.43	50	59.66	49	01	02	20

It can be seen in Table 7.8 and Figure 7.7, that GE has been improved in 100% of the problems. This is a clear indication that allowing the formation of single machine cells is beneficial as far as effective grouping of parts into families and machines into corresponding groups is concerned.

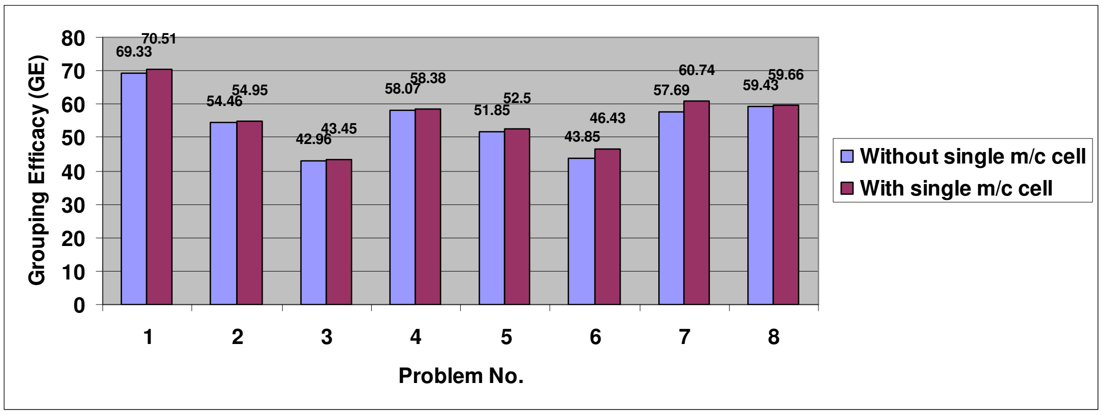


Figure 7.8: Comparison of GE with and without the presence of single machine cells

Increase in GE means: increase in machine utilization inside the cell (less number of 0s inside the block diagonal) and/or decrease in the number of intercellular moves (less number of 1s outside the block diagonal). It can also be seen in Table 7.8

that in addition to increase in GE in 100% problems, a decrease in the total number of intercellular moves is experienced in more than 60% of the problems (5 out of 8). This trend of decrease in intercellular moves is shown in Figure 7.8.

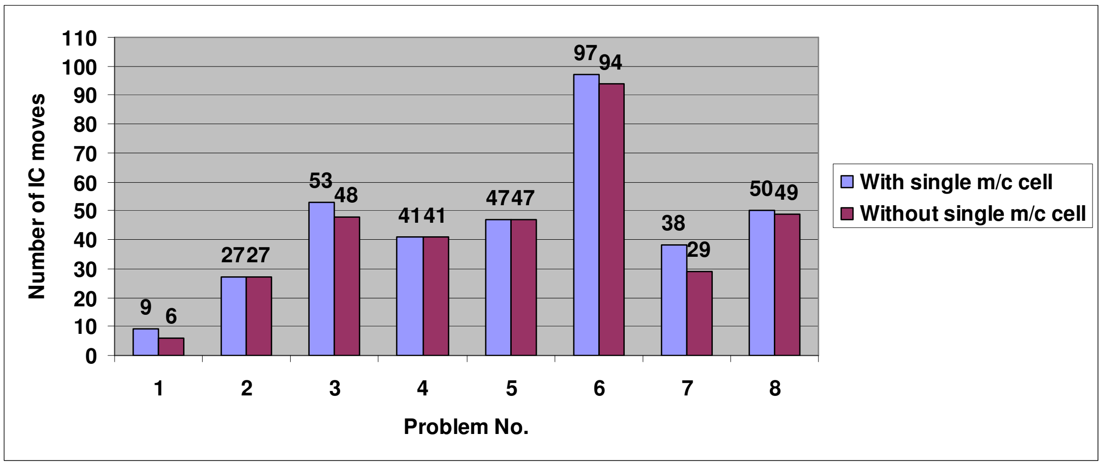


Figure 7.9: Comparison in terms of number of intercellular moves with and without the presence of single machine cells

Reduction in intercellular moves means less material handling cost. This trend in the units of material handling cost saved, as a result of reduction in the total number on intercellular moves while allowing the formation of single machine cells, is shown in Figure 7.9, in the following.

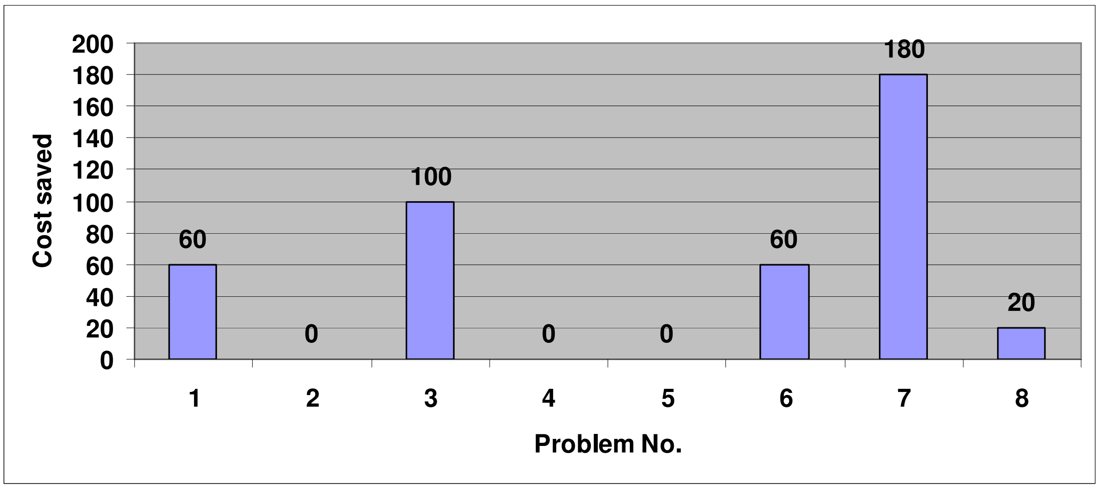


Figure 7.10: Cost saved as a result of reduction in total number of IC moves

7.5 Computational Results and Discussion (Hybrid GA for JSSP): (Tariq et al. [2007])

Table 7.9 presents the results of the tested problems. Though the computational experience is very limited, it shows that in case of more than 90%

problems the optimum result has been achieved. This shows that the proposed algorithm is effective enough as far as its accuracy is concerned.

Table 7.9: Computational Results (Hybrid GA for JSSP) (Tariq et al. [2007])

S/No	**Source**	**Prob**	**Size** ***Jobs × machs***	**Makespan found here**	**Optimal Makespan**	**%age Sol Gap**	**CPU time (sec)**	**Num of Gens**
1	Fisher & Thompson [1963]	FT 6	6 × 6	55	55	0	1	1
2	Lawrence, [1984]	LA 1	10 × 5	666	666	0	4	1
3	Lawrence, [1984]	LA 2	10 × 5	655	655	0	328	36
4	Lawrence, [1984]	LA 3	10 × 5	597	597	0	431	38
5	Lawrence, [1984]	LA 4	10 × 5	590	590	0	13	1
6	Lawrence, [1984]	LA 5	10 × 5	593	593	0	1	1
7	Lawrence, [1984]	LA 6	15 × 5	926	926	0	4	1
8	Lawrence, [1984]	LA 7	15 × 5	890	890	0	12	1
9	Lawrence, [1984]	LA 11	20 × 5	1222	1222	0	59	1
10	Fisher & Thompson [1963]	FT 10	10 × 10	936	930	0.65	6653	90
11	Noor [2007]	CB-JSSP1	8×6	505	505	0	8	2
12	Noor, [2007]	CB-JSSP2	6×6	444	444	0	3	1
13	Noor, [2007]	CB-JSSP3	6×6	379	379	0	18	4

In Table 7.9, comparatively smaller size of problems is solved. The reason is that this approach can handle a maximum problem size (*Machs* × *Parts*) of 100 i.e *Machs* × *Parts* ≤ 100. It is due to the limitation of the software (Applications Manager – AM), this technique has been encoded in. The software (AM) does not allow an array size having more than 8000 elements which means the population size for a 10 × 10 problem cannot be more than 80. This is the reason that as the problem size increases (gets beyond 10 × 10) the size of population gets smaller in order to keep the population array's size (*Machs* × *Parts* × Pop Size) down to 8000 elements (*Machs* × *Parts* × Pop Size ≤ 8000). The decrease in population size means less space for the algorithm to explore, which in turn means that there is every chance for the algorithm to get trapped on a local optimum. A reasonably larger population size provides a larger searching space and also maintains a satisfactory level of diversity. Diversity is important because it keeps the potential areas intact in a population where the global optimum may exist. A smaller population size can get monotonous in comparatively lesser number of generations thus causing the algorithm to get trapped on a local optimum. Now, the question arises that in spite of this limitation why the technique has been encoded in AM? The answer is that we had already encoded the tool for Machine-Part grouping in AM and since this tool for Job-Shop scheduling was to be combined with that, therefore it had to be encoded in AM. Also, at that time this limitation was not known as it did not get discovered during the encoding of Machine-Part grouping technique. Besides that, the reason for selecting AM was that it is user friendly and can be used for quick development of applications.

7.6 Computational Results and Discussion (Combined Model):

As described earlier the combined model is developed by the combination of two tools (hybrid GA for Machine-Part grouping problem and JSSP) in a manner that the output of Machine-Part grouping tool is used as input by the Job-Shop scheduling tool. A number of problems (20) have been selected from literature to evaluate the performance of the Combined model. Results (calculated values of GE and Makespan) of the tested problems are presented in Table 7.10, in the following.

Table 7.10: Results of the problems solved by combined model

S/ No	**Problem size (*Machs*× *Parts*)**	**Source**	**Num of cells**	**Source (GE)**	**This research (*GE*)**	**%age Improvement**	**Processing sequence**	**Processing time**	**Lower bound**	**Makespan C_{max}**	**%age Sol Gap.**
1	5×5	Won & Currie [2006]	2	73.33	73.33	0	Rand	Rand	52	62	19.23
2	5×7	Fernando & Mauricio [2002]	2	73.68	73.68	0	Rand	Rand	59	59	0
3	5×7	Fernando & Mauricio [2002]	2	62.50	69.57	11.31	Rand	Rand	59	68	15.24
4	6×6	Zhao & Wu [2000]	2	76.19	76.19	0	Rand	Rand	79	79	0
5	6×7	Sunderesh & Heragu [1994]	2	65.22	65.22	0	Source	Rand	50	63	26.00
6	6×8	Fernando & Mauricio [2002]	2	76.92	76.92	0	Rand	Rand	113	113	0
7	6×8	etidweb.tamu.edu/ftp/ENTC380/Exam%203%20Material/**18**-Cellular%20Manufacturing.pdf	3	88.89	88.89	0	Rand	Rand	64	64	0
8	6×13	Arzi et al [2001]	2	56.25	56.25	0	Rand	Rand	116	116	0
9	7×8	home.postech.ac.kr/~jjujju/data/data/ch1	3	85.00	85.00	0	Rand	Rand	58	66	13.79

		0(KimJ).ppt									
10	7×9	Yasuda & Yin [2001]	2	72.73	72.73	0	Rand	Rand	64	77	20.31
11	7×10	Mungawattana [2000]	3	69.23	69.23	0	Source	Rand	82	82	0
12	7×11	Fernando & Mauricio [2002]	3	53.13	58.62	10.33	Rand	Rand	77	77	0
13	7×11	Fernando & Mauricio [2002]	3	70.37	70.37	0	Rand	Rand	55	65	18.18
14	7×11	Amirthagadeswaran & Arunachalam, [2006]	2	61.90	61.90	0	Source	Source	55	55	0
15	7×12	Mungawattana [2000]	3	60.00	60.00	0	Source	Rand	76	76	0
16	7×14	Wu et al [2006]	3	65.79	65.79	0	Rand	Rand	77	77	0
17	8×10	Murugan & Selladurai, [2005]	3	81.25	81.25	0	Source	Rand	74	87	17.56
18	8×12	Fernando and Mauricio [2002]	3	68.30	68.30	0	Rand	Rand	102	102	0
19	9×9	Gongaware & Ham [1981]	3	74.29	74.29	0	Rand	Rand	82	90	9.75
20	10×10	Fernando and Mauricio [2002]	3	70.59	70.59	0	Rand	Rand	56	72	28.57

Table 7.10 shows that the Combined model not only provides solution for the Machine-Part grouping problem but also provides solution for the system scheduling problem. In the part machine grouping stage only two problems (3 & 12) have been further improved as far as their GE is concerned. Whereas, in the scheduling part eleven problems (55% of the tested problems) achieved the lower bound values which shows the effectiveness of the scheduling tool. A graphical representation of the %age solution gap between the lower bound and the Makespan achieved, for each problem, by the hybrid GA based scheduling tool developed during this research is given in Figure 7.11.

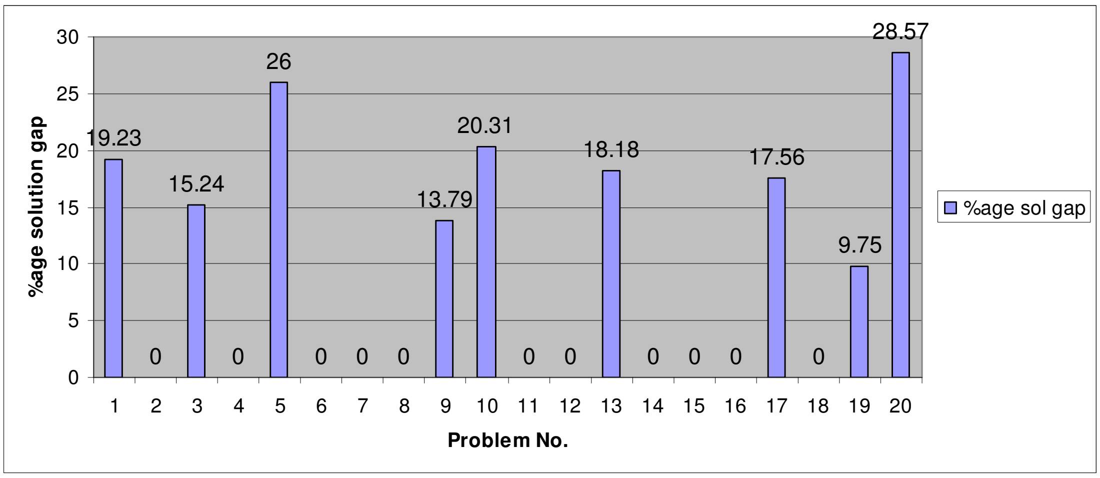


Figure 7.11: %age solution gap between LB and result achieved for each problem

It can also be seen in the Table 7.10 that for 75% of the problems (15 out of 20) the processing sequence has been randomly generated as it is not provided in the source. Similarly, for 95% of the problems (19 out of 20) the processing times have been randomly generated. Whereas for none of the above mentioned problems, the values of Makespan are available in literature. This shows that this kind of solution approach is very rare to be found in literature as most of the researchers emphasized heavily on Machine-Part grouping and little attention has been paid to operational issues.

7.7 Summary:

This chapter has presented a detailed analysis based on the testing and validation of the tools developed for Machine-Part grouping and Job-Shop scheduling. A number of benchmark problems have been selected from literature and after solving the results have been compared with the already reported results by different researchers. The results show that the Machine-Part grouping tool, besides performing satisfactorily has also one other aspect of allowing the formation of single machine cells which results in increasing GE while reducing the number of intercellular moves and corresponding material handling cost. On the other hand, though the tool developed for JSSP performed efficiently for the tested problems, listed in Table 6.9, but it still carries a drawback as it could not be applied to large size problems because of the limitations (on the size of array) of the software (AM) used for encoding. Finally, after separate testing and validation of the two tools (Machine-Part grouping problem and JSSP) they have been combined and then the combined model is also tested and validated through a number of problems from literature. The problem with validation of the combined model is that benchmark results for the scheduling part of the problem are not available in literature, showing that this aspect of CMS design (operational) has not been thoroughly considered by researchers.

CHAPTER 8

CONCLUSION AND FUTURE RESEARCH

8.1 Introduction:

The main research objectives initially described in Chapter 1 were to develop separate hybrid tools for the Machine-Part grouping and Job-Shop scheduling problems and then finally integrate them together after verification and validation of the tools through a number of benchmark problems form literature. Details regarding the development of these tools, their integration, and validation are described in Chapter 5, Chapter 6 and Chapter 7. Now, this chapter presents the outcome of the technique developed during this research for the operational design of a CMS. In addition to the overall conclusion of this research some directions for the future work in this area are also identified.

8.2 Conclusion of Hybrid GA for Machine-Part Grouping:

The hybrid GA based approach, for Machine-Part grouping, developed during this research, combines an LSH with GA that uses integer type representation, multipoint crossover and roulette wheel selection procedure. LSH is placed inside the GA loop in such a way that the best solution in each generation is subjected to it. After being locally improved the solution is placed back in population so that it can take part in different genetic operators like crossover and mutation and produce even better solutions. A computational experience (Table 7.1 & Table 7.2) shows that the approach developed during this research is not only accurate, as it produce results that are equally accurate, but also achieves this goal in comparatively lesser number of generations. Further, in case of some problems (41.67% of the total tested problems) the values of GE obtained by this approach are greater than the ones reported earlier as best. Another differentiating feature of the approach is that it has shown a consistent level of accuracy in case of all problem sizes whether small or large. On the contrary other approaches reported in literature and used here as benchmarks have the tendency to loose their accuracy and effectiveness with the increase in problem size and the ill-structured formation of Machine-Part incidence matrices. A statistical analysis has also been carried out to further verify and authenticate the effectiveness of this approach.

Another important aspect of this technique is that, unlike many other approaches, here the formation of cells that posses one machine only is allowed. The computational experience (Table 7.7 & Table 7.8), as far as this aspect of the algorithm is concerned, shows that by allowing the formation of cells that contain only one machine a decrease can be observed in the total number of intercellular moves and corresponding material handling costs which results an increase in the value of GE.

8.3 Conclusion of Hybrid GA for Job-Shop Scheduling:

Similar to the approach developed for Machine-Part grouping a hybrid GA based technique is developed for Job-Shop scheduling by combining GA with an LSH while representing chromosomes as two-dimensional arrays of integers, making use of the swap mutation, multi-cut point crossover, and the selection approach of Stochastic Universal Sampling (SUS). The uniqueness of the approach, that was initially intended, is that each chromosome (solution) is represented in the form of a two dimensional array which is very rare to be found in literature. Also, a repair algorithm is developed that ensures the legality of each solution in a population. Here also, the LSH developed during this research is placed on the inner side of the GA loop and locally improves the best solution of each generation. The main significance of the LSH developed during this research is that it produces satisfactory results in combination with ordinary two point crossover and swap mutation. It means that the LSH is so effective that it has relieved considerable amount of pressure on the GA operators (crossover, mutation and selection) which is the main reason for satisfactory results. Computational experience (Table 7.9) with the algorithm shows that in more than 90% (12 out of 13) of the problems the results obtained are optimum. Also, in 77% (10 out of 13) of the total tested problems the algorithm consumed a maximum of four generations in reaching the optimum result which proves the effectiveness of the LSH which helps the algorithm to converge on to the optimum in lesser number of generations.

Though the technique developed during this research seems effective, as far as the reported results are concerned, but it still has one major shortcoming. This approach can handle a maximum problem size (*Machs* × *Parts*) of 100 i.e *Machs* × *Parts* ≤ 100. It is due to the limitation of the software (Applications Manager – AM), this technique has been encoded in. The software (AM) does not allow an array size greater than 8000 elements that means the population size for a 10 × 10 problem cannot be more than 80. This is the reason that as the problem size increases (gets beyond 10 × 10) the size of population gets smaller in order to keep the population array's size (*Machs* × *Parts* × Pop Size) down to 8000 elements (*Machs* × *Parts* × Pop Size ≤ 8000). The decrease in population size means less space for the algorithm to explore.

8.4 Conclusion of Combined Model (Operational Design of a CMS):

The main motivation for this research was to develop a combined technique that not only provides solution for the Machine-Part grouping problem (cell design) but also handles the system scheduling problem which is unique in itself as both the problems have been dealt with separately in literature. After separate development and thorough validation of hybrid GA based approaches both for Machine-Part grouping and Job-Shop scheduling the two models have been combined with each other in such a way that the output of Machine-Part grouping tool is used as an input to the Job-Shop scheduling tool. The uniqueness of this model, as already described, is that it not only groups machines into cells and parts into corresponding families but also provides solution for the system's scheduling problem. This combined solution approach is very rare to be found in literature and thus makes this technique different and more useful in practice as far as other competitive existing techniques, in literature, are concerned. A reasonable computational experience presented in Table 7.10 shows that such problems have not been handled previously and that is the reason that for none of the problems solution both in terms of GE and Makespan is available. In 10 % of the problems (2 out of 20) the GE obtained is better than the previously reported results, whereas in 90% of the problems (18 out of 20) the GE obtained is equal to the previous best results. In 75% of the total tested problems (15 out of 20) the processing sequence has been randomly generated as it is not provided in the source. Similarly, for 95% of the problems (19 out of 20) the processing times have been randomly generated. Whereas for none of the above mentioned problems, the values of Makespan are available in literature. This shows that this kind of solution approach is very rare to be found in literature as most of the researchers emphasized heavily either on Machine-Part grouping or on system scheduling.

8.5 Directions for Future Research:

Though the techniques developed during this research both for machine part grouping and Job-Shop scheduling performed satisfactorily but still there is a lot of room for improvement. In the following sections identification and description of certain directions for future research are presented which can further diversify and improve the working of these techniques.

8.5.1 Hybrid GA for Machine-Part grouping:

Following are some of the directions for future research that may lead to improving the hybrid GA based tool developed for the grouping machines into cells and parts into their corresponding families.

1. Although the developed methodology for the design of CMS produced satisfactory results (better than the reported results in literature) but the approach was based on static conditions where it is assumed that the parts are already available at time zero. It is proposed that in future the dynamic conditions should also be considered to come up with a more realistic operational design of the CMS. Further it is suggested that demand, processing times, etc. should also be considered while designing a CMS as opposed to just clustering, partitioning.
2. In order to take advantage of the available GA tool box in MATLAB and to overcome the shortcomings of the AM software this technique is proposed to be encoded in MATLAB.
3. Other performance measures such as minimizing: material handling cost, work in process etc may also be included to evaluate the quality of the solution.
4. To make this technique more time efficient the LSH is proposed to be made more intelligent by identifying and excluding some of the checks which can not improve GE any further. For example; by changing a gene's value which results in placing the corresponding part or machine into a cell where it doesn't have any operation will not improve GE. LSH can be made intelligent to identify such checks in advance and avoid wasting time in running the whole procedure for such options.

8.5.2 A Hybrid GA for Job-Shop Scheduling:

Some directions for future research to improve the working and performance of the hybrid GA based tool for JSSP are presented in the following:

1. It has already been pointed out that the major deficiency in the tool developed during this research for solving the JSSP is its inability to handle large size problems which is due to the limitations of the software (Applications Manager – AM) it has been encoded in. Therefore it is suggested that the same technique may be encoded in MATLAB software to overcome this difficulty (array size limitation).
2. It is further suggested that different types of crossover and mutation operators may be implemented for further improvement in performance.
3. LSH is suggested to be made more intelligent to avoid such checks which may not be able to improve results. This would make the technique more time efficient.
4. The current system is a two-part sequential system in which the first part deals with the cell formation and the second part finds a schedule for the operations carried out in the cell. The results of scheduling module are not used to modify the original cell formation. Therefore, it is suggested that the scheduling part of the algorithm should be Combined with the cell design so that information about the overall Makespan can be used as a feed back and the initial cell design (Machine-Part grouping) can be modified for further improvement in the overall Makespan. This would be a further improvement as far as integration of the model is concerned.
5. Other performance measures such as flow time minimization, earliness, tardiness etc. may also be considered to check the quality of the solution.

8.6 Summary:

The primary and secondary objectives set for this research have been successfully achieved. A hybrid GA based tool for Machine-Part grouping has been developed and its performance has been validated through a number of benchmark problems from literature. Though this tool has performed satisfactorily but still to further improve its performance, some recommendations have been made for future work. Another hybrid tool to handle the system scheduling problem has also been developed. The performance of this tool has also been tested and validated through a number of benchmark problems from literature. Further, the shortcomings of the tool have been identified, and to overcome those some directions for future research have also been described.

References:

[1]. Aaron L.N., Moustapha D., Wilson L.P. [2006]. *"Hybrid genetic approach for solving large scale capacitated cell formation problems with multiple routings"*. European Journal of Operational Research, 171(3), 1051-1070.

[2]. Aarts, E. H. L., and Van Laarhoven, P. J. M., [1985]. *"Statistical cooling: approach to combinatorial optimization problems"*. Philips Journal of Research, 40, 193-226.

[3]. Akers S. B., [1954]. *"A Graphical Approach to Production Scheduling Problems"*. Operations Research, 4, 244-245.

[4]. Akturk, M.S., and Turkcan, [2000]. *"A Cellular Manufacturing System design using a holonistic approach"*. International Journal of Production Research, 38(1) 2327-2347.

[5]. Alfa,A.,Heragu, S., and Chen,M., [1991]. *"A 3-opt based simulated annealing algorithm for vehicle routing problems"*. Computers and Industrial Engineering, 21, 635-639.

[6]. Amirthagadeswaran, K. S., & Arunachalam, V. P., [2006]. *"Optimization of Scheduling in Job Shops under Cellular Manufacturing Environment using Genetic Algorithm"*. Journal of The Institution of Engineers (India), 86, 39-42.

[7]. Application Manager (AM), [2001]. Intelligent Environment, Middlesex, UK.

[8]. Al-Qattan, I., [1990]. *"Designing flexible manufacturing cells using branch and bound method"*. International Journal of Production Research, 28(2), 325-336.

[9]. Al-Sultan, K.S. & Fedjki, C.A., [1997]. *"A genetic algorithm for the part family formation problem"*. Production Planning & Control, 8, 788–796.

[10]. Annan, M. [2000]. *"Design of Cellular Manufacturing Systems for Dynamic & Uncertain Production Requirements with Presence of Routing Flexibility"*. Ph.D. Thesis.

[11]. Applications Manager (AM), [2001]. *"Intelligent Environment"*, Middlesex, UK.

[12]. Arizono, I., Yamamoto, A., Ohta, H., [1992]. *"Scheduling for minimizing total actual flow time by neural networks"*. International Journal of Production Research, 30(3), 503 – 511.

[13]. Arzi, Y., Bukchin, J. and Masin, M., [2001]. *"An efficiency frontier approach for the design of Cellular Manufacturing Systems in a lumpy demand environment"*. European Journal of Operations Research, 134, 346–364.

[14]. Asano M. and Ohta H., [2002]. *"A Heuristic for Job Shop Scheduling to Minimize Total Weighted Tardiness"*. Computer and Industrial Engineering, 42, 137-147.

[15]. Askin, R., & Chu, K. [1990]. *"A Graph Partitioning Procedure for Machine Assignment and Cell formation in Group Technology"*. International Journal of Production Research, 28(8), 1555-1572.

[16]. Askin, R. G., & Standridge, C. R., [1993]. *"Modelling and Analysis of Manufacturing System"*. Wiley, New York.

[17]. Askin, R. G. & Subrammanian, S. [1987]. *"A cost-based heuristic for group technology configuration"*. International Journal of Production Research, 25(l), 101-113.

[18]. Balas E., [1965]. *"An Adictive Algorithm for Solving Linear Programming with Zero-One Variables"*. Operation Research, 13, 517-546.

[19]. Balas E., [1969]. *"Machine Scheduling Via Disjunctive Graphs: An Implicit Enumeration Algorithm"*. Operation Research, 17, 941-957.

[20]. Balas E., [1979]. *"Dynamic Programming"*. Hammer P. L., Johson E. L. and Korte B. (Eds), Discrete Optimization II, North Holland, Amsterdam, The Netherlands.

[21]. Ballakur, A. & Steudel, H. J., [1987]. *"A Within-Cell Utilization Based Heuristic for Designing Cellular Manufacturing Systems"*. International Journal of Production Research, 25(5), 639-665.

[22]. Barton, J. & Love, D., [2005]. *"Retrieving designs from a sketch using an automated GT coding and classification system"*. Production Planning & Control, 16(8), 763–773.

[23]. Baykasoglu, A., Gindy, N.N.Z., [2000]. *"MOCACEF 1.0: Multiple objective capability based approach to form part–machine groups for Cellular Manufacturing applications"*. International Journal of Production Research, 38, 1133–1161.

[24]. Bedworth D., Bailey J. E., [1987]. *"Integrated Production Control Systems, Management Analysis, Design"*. 2nd Edition, John Wiley & Sons.

[25]. Bellman R., [1956]. *"Dynamic Programming and Lagrange Multipliers"*. Proceedings of the National Academy of Sciences, 42, 767-769.

[26]. Ben-Arieh, D., Sreenivasan, R., [1999]. *"Information analysis in a distributed dynamic group technology method"*. International Journal of Production Economics, 60–61, 427–432.

[27]. Bezdek, J.C., [1981]. *"Pattern Recognition with Fuzzy Objective Function Algorithm"*. New York, Plenum Press.

[28]. Bierwirth C., [1995]. *"A Generalized Permutation Approach to Job Shop Scheduling with Genetic Algorithms"*. OR Spektrum, 17(2-3), 87-92.

[29]. Bierwirth, C., Mattfield, D. C., Kopfer, H., [1996]. *"A generalized permutation approach to Job-Shop scheduling with genetic algorithms"*. OR Spectrum, 17(2-3), 87-92.

[30]. Billo, R. E., [1999]. *"Organizing principles for the design of classification and coding software"*. Journal of Manufacturing Systems, 17(6), 405-417.

[31]. Black, J. [1983]. *"Cellular Manufacturing Systems Reduce Setup Time, Make Small Lot Production Economical"*. Industrial Engineering, 36-48.

[32]. Black, J. [1991]. *"The Design of the Factory with a Future"*. McGraw Hill Inc. New York.

[33]. Blazewicz J., Dror M. and Weglarz J., [1991]. *"Mathematical Programming Formulations for Machines Scheduling: A survey"*. European Journal of Operational Research, Invited Review, 51(3), 283-300.

[34]. Boctor, F.F. [1991]. *"A linear formation of the machine cell formation problem"*. International Journal of Production Research, 29(2), 343-356.

[35]. Boctor, F.F., [1996]. *"The minimum-cost, Machine-Part cell formation problem"*. International Journal of Production Research, 34, 1045–1063.

[36]. Boe, W. J. & Cheng, C. H. [1991]. *"A close neighbour algorithm for designing Cellular Manufacturing System"*. International Journal of Production Research, 29(10), 2097-2116.

[37]. Bonomi, E. & Lutton, J., [1984]. *"The N-city travelling salesman problem: Statistical mechanics and the Metropolis algorithm"*. SIAM Review, 26, 551-568.

[38]. Burbidge, J.L. [1975]. *"The Introduction of Group Technology"*. Wiley, New York.

[39]. Carpenter, G. A., & Grossberg, S., [1987]. *"A massively parallel architecture for a self organizing neural pattern recognition machine"*. Computer Vision, Graphics and Image Processing, 37, 54-115.

[40]. Carpenter, G. A., & Grossberg, S., [1991]. "

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

[42]. Caux, C., Bruniaux R., & Pierreval, H., [2000]. *"Cell formation with alternative process plans and machine capacity constraints: A new combined approach"* International Journal of Production Economics, 641(1-3), 279-284.

[43]. Celano, G., Costa, A. & Fichera, S. [2007]. *"Scheduling of unrelated parallel manufacturing cells with limited human resources"*. International Journal of Production Research, 46(2), 405-427.

[44]. Chandrasekharan, M. P., & Rajagopalan, R. [1986a]. *"An ideal seed non-hierarchical clustering algorithm for cellular manufacturing"*. International Journal of Production Research, 24(2), 451-464.

[45]. Chandrasekharan, M. P., & Rajagopalan, R. [1986b]. *"MODROC: An extension of rank order clustering for group technology"*. International Journal of Production Research, 24(5), 1221-1233.

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

[47]. Chandrasekharan, M. P., & Rajagopalan, R. [1989]. *"GROUPABILITY: Analysis of the properties of binary data matrices for group technology"*. International Journal of Production Research, 27(6), 1035-1052.

[48]. Chan, H. M. & Milner, D. A. [1982]. *"Direct clustering algorithm for group formation in cellular manufacture"*. Journal of Manufacturing System, 1, 65-75.

[49]. Chaperfield A., Flemming P., Pohlhein H., Fonseca C., [2001]. *"Genetic Algorithm MATLAB Tool Box –User's Guide"*. Version 1.2, Department of Automatic Control and Systems Engineering, University of Sheffield.

[50]. Chen, S. J., & Cheng, C. S., [1995]. *"A neural network based cell formation algorithm in cellular manufacturing"*. International Journal of Production Research, 33(2), 293-318.

[51]. Chen, C.L., Cotruvo, N.A. and Bake, W., [1995]. *"A simulated annealing solution to the cell formation problem"*. International Journal of Production Research, 33, 2601–2614.

[52]. Chen, J. and Heragu, S.S., [1999]. *"Stepwise decomposition approaches for large scale cell formation problems"*. European Journal of Operational Research, 11, 64-79.

[53]. Chen, W.H., & Srivastava, B., [1994]. *"Simulated annealing procedures for forming machine cells in group technology"*. European Journal of Operational Research, 75, 100-111.

[54]. Cheng, C.H., Gupta, Y.P., Lee, W.H. & Wong, K.F. [1998]. *"A TSP-based heuristic for forming machine groups and part families"*. International Journal of Production Research. 36(5), 1325-1337.

[55]. Cheng, C. H., [1995]. *"A branch and bound clustering algorithm"*. IEEE Transactions on Systems, Man, and Cybernetics, 25(5), 895-898.

[56]. Cheng-Fa Tsai, Feng-Cheng Lin, [2003]. *"A new hybrid heuristic technique for solving job shop scheduling problem"*. IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, Lviv, Ukraine.

[57]. Cheng R., Mitsuo G. and Tsujimura Y., [1996]. *"A tutorial survey of Job-Shop scheduling problems using genetic algorithms-I –Representation"*. Computer and Industrial Engineering, 30(4), 983-997.

[58]. Cheng R., Mitsuo G. and Tsujimura Y., [1999]. *"A tutorial survey of Job-Shop scheduling problems using genetic algorithms-part II: Hybrid genetic search strategies"*. Computer and Industrial Engineering, 36, 343-364.

[59]. Choobineh, F. [1988]. *"A Framework for the Design of Cellular Manufacturing Systems"*. International Journal of Production Research, 26(7), 1161-1172.

[60]. Chu, C. H., Hayya, J. C., [1991]. *"A fuzzy clustering approach to manufacturing cell formation"*. International Journal of Production Research, 29(7), 1475-1487.

[61]. Chu, C.H., & Tsai, M. [1990]. *"A Comparison of Three Array-Based Clustering Techniques for Manufacturing Cell Formation"*. International Journal of Production Research, 28(8), 1417-1433.

[62]. Dagli, C. & Huggahalli, R., [1995]. *"Machine-Part family formation with the adaptive resonance theory paradigm"*. International Journal of Production Research, 33, 893-913.

[63]. Della C. F., Tedei R., Rolando R., [1994]. *"Solving a Real World Project SchedulingProblem with a Genetic Approach"*. Belgian Journal of Operations Research, Statistics and Computer Science, 33(1-2), 65-78.

[64]. Della C. F., Tadei R., Volta G., [1995] *"A genetic algorithm for Job-Shop problem"*. Computers and Operations Research, 22(1), 15-24.

[65]. Devis L., [1985]. *"Applying Adaptive Algorithms to Epistatic Domains"*. Proceedings of the International Joint Conference on Artificial Intelligence, 162-164.

[66]. Dimopoulos, C., & Mort, N., [2001]. *"A hierarchical clustering methodology based on genetic programming for the solution of simple cell-formation problems"*. International Journal of Production Research, 39(1), 1–19.

[67]. Dorndorf U., Pesch E., [1995]. *"Evolution based Learning in Job Shop Scheduling Environment"*. Computers and Operation Research, 22(1), 25-40.

[68]. Edwards D., Taylor N., Brown K., [2001]. *"Comprehensive Evolution of Neural Networks"*. Proc of the 2001 UK Workshop of Computational Intelligence, University of Edinburgh, 75-80.

[69]. *etidweb.tamu.edu/ftp/ENTC380/Exam%203%20Material/**18**-Cellular%20Manufacturing.pdf* – (Problem 7, Table 6.10)

[70]. Falkenaur E., Boufoix S., [1991]. *"A Genetic Algorithm for Job Shop"*. Proceedings of the IEEE International Conference on Robotics and Automation, 824-829.

[71]. Fang H. L., Ross P. and Corne D, [1993]. *"A Promising Genetic Algorithm Approach to Job Shop Scheduling, Rescheduling and Open Shop Scheduling Problems"*. ICGA's Proceedings of the 5th International Conference on Genetic Algorithms, Morgan Kaufmann Publishers, San Mateo, California, 375-382.

[72]. Fang H. L., Ross P. and Corne D, [1994]. *"A promising hybrid GA/Heuristic approach for Open Shop Scheduling Problems"*. Proceedings of the 11th European Conference on Artificial Intelligence, 590-594.

[73]. Ghedjati, F., [1999]. *"Genetic algorithms for the Job-Shop scheduling problem with unrelated parallel constraints: Heuristic mixing method*

machines and precedence". Computers & Industrial Engineering 37(1-2), 39-42.

[74]. Fernando, J. G. & Mauricio, G. C. R., [2002]. "*A hybrid genetic algorithm for manufacturing cell formation. Technical Report".* TD-5FE6RN. AT&T Labs Research.

[75]. Fisher H., Thompson G. L., [1963]. *"Probabilistic learning combinations of local job shop scheduling rules",* Muth, J. F., Thompson G. L.(Eds), Industrial Scheduling, Prentic-Hall, Englewood Cliffs, NJ.

[76]. Florian M., Trepant P. and Mohan G. B., [1971]. *"An implicit Enumeration Algorithm for Machine Sequencing Problem".* Management Science Application Series, 17(12), 782-792.

[77]. Fogarty D.W., Blackstone J.H. & Hoffmann T.R., [1991]. *"Production and Inventory Management".* South –Western Publishing Co.

[78]. Foo, S. Y., & Takefuji, Y., [1988]. *"Integer linear programming neural networks for Job-Shop scheduling".* Proceedings of the IEEE International Conference on Neural Networks, San Diego, California, USA, 2, 341-348.

[79]. Frazier, G. V., Gaither, N., & Oslon, D., [1990]. *"A procedure for dealing with multiple objectives in cell formation decisions".* Journal of Operations Management, 9(40), 465-480.

[80]. Fry, T. Breen, M. & Wilson, M. [1987]. *"A Successful Implementation of Group Technology & Cellular Manufacturing".* Production & Inventory Management Journal, 28(3): 4-6.

[81]. Gary, M. R., & Johnson, D. S., [1979]. *"Computers and intractability: A guide to the theory of NP-Completeness".* Freeman.

[82]. Gen, M., & Cheng, R. [1997]. *"Genetic Algorithms & Engineering Design".* John Wiley & Sons, Inc.

[83]. Gen M., Tsujimura Y., Kubota E., 1994]. *"Solving Job Shop Scheduling Problem Using Genetic Algorithms".* Gen M., (Ed), Proceedings of the 16th International Conference on Computer and Industrial Engineering, Japan, 576-579.

[84]. Geoffrey, O. O., Chen, M., Chanagchit, C., & Richard L. S., [1992]. *"Manufacturing system cell formation and evaluation using a new inter-cell reduction heuristic".* International Journal of Production Research, 30(5), 1101-1118.

[85]. Geonwook, J. & Herman, R. L. [2006]. *"Forming part families by using genetic algorithms and designing machine cells under demand changes"*. Computers & Operations Research, 33, 263-283.

[86]. Geonwook, J., Herman, R. L., & Hamid, R. P., [1998]. *"A Cellular Manufacturing System based on new similarity coefficient which considers alternative routes during machine failure"*. Computers and Industrial Engineering, 34(1), 21-36.

[87]. Geonwook, J., Herman, R. L., [2006]. *"Forming part families using Genetic Algorithm and designing machine cells under demand changes"*. Computers and Research, 33, 263-283.

[88]. Giffler B. and Thompson G. L., [1960]. *"Algorithms for Solving Production Scheduling Problems"*. Operations Research, 8(4), 487-503.

[89]. Gill, A., Bector, C. R., [1997]. *"A fuzzy linguistic approach to data quantification and construction of distance measures for the part family formation problem"*. International Journal of Production Research, 35, 2565-2578.

[90]. Goldberg, D.E., [1989]. *"Genetic Algorithms in Search, Optimisation, and Machine Learning"*. Addison-Wesley: New York, NY.

[91]. Goldberg D., Lingle R., [1985]. *"Alleles, Loci and the Travelling Salesman Problem"*. Proceedings of the First International Conference on Genetic Algoirthms, Hillsdale, NJ:Lawrence Erlbaum Associates, 154-159.

[92]. Gongaware, & I. Ham, I., [1981]. *"Cluster analysis applications for group technology manufacturing systems"*. Proceedings of the 9th North American Manufacturing Research Conference (NAMRC), 503-508.

[93]. Gravel, M., Nsakanda, A.L. and Price, W., [1998]. *"Efficient solutions to the cell-formation problem with multiple routings via a double-loop genetic algorithm"*. European Journal of Operations Research, 109, 286–298.

[94]. Greene, T. and Sadowski, R., [1984]. *"A Review of Cellular Manufacturing Assumptions, Advantages, and Design Techniques"*. Journal of Operations Management, 4(2), 85-97.

[95]. Groover, M. P., [2008]. *"Automation production system and computer integrated manufacturing"*. Prentice Hall, New Jersey, USA.

[96]. Gunasingh, K. R., & Lashkari, R. S., [1989]. *"The Cell Formation Problem in Cellular Manufacturing Systems-A Sequential Modeling Approach".* Computers and Industrial Engineering, 16(4), 469-476.

[97]. Gunasingh, K. R., & Lashkari, R. S., [1989]. *"Machine Grouping in Cellular Manufacturing Systems--An Integer Programming Approach".* International Journal of Production Research, 27(9), 1465-1473.

[98]. Gunasingh, K. R., & Lashkari, R. S., [1990]. *"Simultaneous grouping of parts and machines in Cellular Manufacturing Systems: an integer programming approach".* Computers and Industrial Engineering, 20(1), 111-117.

[99]. Gupta, T., Seifoddini, H., [1990]. *"Production data based similarity coefficient for machine component grouping decisions in the design of a Cellular Manufacturing System".* International Journal of Production Research, 28(7), 1259-1273.

[100]. Gupta, Y., Gupta, M., Kumar, A., & Sundram, C., [1995]. *"Minimizing total intercell and intracell moves in cellular manufacturing: A genetic algorithm approach".* International Journal of Computer Integrated Manufacturing, 8(2), 92-101.

[101]. Gupta, Y., Gupta, M., Kumar, A. and Sundaram, C., [1996]. *"A genetic algorithm-based approach to cell composition and layout design problems".* International Journal of Production Research, 34, 447–482.

[102]. Hagan M. T., Demuth H. B., Beale M., [2002]. *"Neural Network Design".* Vikas Publishing House Pvt. Ltd., New Delhi.

[103]. Ham, I. Hitomi, K. & Yoshida, T. [1985]. *"Layout Planning for Group Technology in Group Technology".* Applications to Production Management, 153-169.

[104]. Harhalakas, G., Nagi, R., & Proth, J. M., [1990]. *"An efficient heuristic in manufacturing cell formation for group technology applications".* International Journal of Production Research, 28(1), 185-198.

[105]. Harjunkoski I., Jain V., and Grossmann I. E., [2000]. *"Hybrid Mixed Integer/Constraint Logic Programming Strategies for solving Scheduling and Combinatorial Optimization Problems".* Computers and Chemical Engineers, 24, 337-343.

[106]. Hefetz N. and Adiri I., [1982]. *"An Efficient Optimum Algorithm for Two Machines Unit Time Job Shop Schedule Length Problem"*. Mathematics of Operation Research, 7, 354-360.

[107]. Heragu, S.S., Chen, J., [1997]. *"Optimal solution of Cellular Manufacturing System design: Benders' decomposition approach"*. European Journal of Operational Research.

[108]. Hitomi K., [1996]. *"Manufacturing Systems engineering – A unified approach to manufacturing technology, production management, and industrial economics"*. Taylor and Francis Ltd, London UK, 2nd edition.

[109]. Holland, J.H., [1975]. *"Adaptation in Natural and Artificial System"*. University of Michigan Press: Ann Arbor, MI.

[110]. home.postech.ac.kr/~jjujju/data/data/ch10(KimJ).ppt (Problem 9, Table 6.10)

[111]. Houtzeel, A., & Brown, C. S., [1984]. *"A Management Overview of Group Technology"* in Group Technology At Work. N. L. Hyer (Ed.), Society of Manufacturing Engineers, Detroit, MI.

[112]. Hwang, H. and Sun, J., [1996]. *"A genetic-algorithm-based heuristic for the GT cell formation problem"*. Computers & Industrial Engineering, 30(4), 941–955.

[113]. Hyer, N. L., Wemmerlov, U. [1989]. *"Group Technology in the US manufacturing industry: A survey of current practices"*. International Journal of Production Research, 27(2), 1287-1304.

[114]. Irani, S. A. [1999]. *"Handbook of Cellular Manufacturing Systems"*. John Wiley & Sons, Inc.

[115]. Jain A.S., [1998]. *"A Multi-Level of Hybrid Framework for the deterministic Job- Shop scheduling problem"*. PhD Thesis, University of Dundee.

[116]. Jain A.S., Meeran S., [1998]. *"Job-Shop scheduling using neural networks"*. International Journal of Production Research, 36(5), 1249-1272.

[117]. Jackson J. R., [1956]. *"An Extension of Johson's Rule on Job Lot Scheduling"*. Naval Research Logistics, 3(3), 201-203.

[118]. Johnson S. M., [1954]. *"Optimal Two and Three Stage Production Schedules with Setup Time Included"*. Naval Research Logistics Quarterly, 1, 61-68.

[119]. Joines, J.A., Culbreth, C.T. & King, R.E. [1996]. *"Manufacturing cell design: An integer programming model employing genetic algorithms"*. IIE Transactions, 28, 69-85.

[120]. Josien, K. and Liao, T.W., [2000]. *"Integrated use of fuzzy c-means and fuzzy KNN for GT part family and machine cell formation"*. International Journal Production Research, 38(15), 3513–3536.

[121]. Kao, Y., & Moon, Y. B., [1990]. *"Learning part families by back propagation rule of neural networks"*. Proceedings of 1st International Conference on Automation Technology, Hsinchu, Taiwan, 819-824.

[122]. Kao, Y., & Moon, Y. B., [1991]. *"A unified group technology implementation using the back propagation learning rule of neural networks"*. Computers and Industrial Engineering, 20(4), 425-437.

[123]. Kaparthi, S., Suresh, N.C. and Cerveny, R.P., [1993]. *"An improved neural network leader algorithm for part-machine grouping in group technology"*. European Journal of Operations Research, 69(3), 342–356.

[124]. Kaparthi, S. & Suresh, N. C. [1990]. *"Machine-component cell formation in group technology: A neural network approach"*. International Journal of Production Research, 30(6), 1353-1367.

[125]. Kaparthi, S., & Suresh, N.C. [1991]. *"A Neural Network System for Shape-Based Classification and Coding of Rotational Parts"*. International Journal of Production Research, 29(9), 1771 -1784.

[126]. Kaparthi, S., & Suresh, N.C. [1992]. *"Machine component cell formation in group technology: A neural network approach"*. International Journal of Production Research, 25(6), 1353-1367.

[127]. Kiang, M.Y., Kulkarni, U.R. and Tam, K.Y., [1995]. *"Self-organising map network as an interactive clustering tool-an application to group technology"*. Decision Support System, 15(4), 351–374.

[128]. Kirkpatrick, S., Gelatt, C. D. & Vecchi, M. P., [1983]. *"Optimization by Simulated Annealing"*. Science, 220, 671-680.

[129]. King, J. R. [1980]. *"Machine-component grouping in production flow analysis; An approach using Rank Order Clustering algorithm"*. International Journal of Production Research, 18(2), 213-232.

[130]. King, J.R. & Nakornchai, V. [1982]. *“Machine-component group formation in group technology: Review and extension”.* International Journal of Production Research, 20(2), 117-133.

[131]. Kobayashi S.,Ono I., Yamamura M., [1995]. *“An Efficient Genetic Algorithm for Job Shop Scheduling Problems”.* Proceedings of the Sixth International Conference on Genetic Algorithms, San Francisco, CA, Morgan Kaufmann Publishers, 506-511.

[132]. Kohonen, T., [1982]. *“Self-Organized Formation of Topologically Correct Feature Maps”.* Biological Cybernetics, 43(1), 56-69.

[133]. Kumar, K. R., & Chandrasekharan M. P., [1990]. *“Grouping efficacy: a quantitative criterion for goodness of block diagonal form of binary matrices in group technology”.* International Journal of Production Research, 28(2), 233-243.

[134]. Kumar, K. R., Kusiak, A. & Vannelli, A. [1986]. *“Grouping of Parts and Components in Flexible Manufacturing Systems*”. European Journal of Operations Research, 24, 387-397.

[135]. Kumar, K. R. and Vannelli, A., [1987]. *“Strategic subcontracting for efficient disaggregated manufacturing”.* International Journal of Production Research, 25(12), 1715-1728.

[136]. Kusiak, A., [1985]. *"The Part Family Problem in Flexible Manufacturing Systems".* Annals of Operations Research, 279-300.

[137]. Kusiak, A. [1987]. *“The Generalized Group Technology Concept”.* International Journal of Production Research, 25(4), 561-569.

[138]. Kusiak, A. & Chow, W. [1987]. *“Efficient solving of the group technology problem”.* Journal of Manufacturing Systems, 6(2), 117-124.

[139]. Kusiak, A. [1988]. *“A knowledge based system for group technology”.* International Journal of Production Research, 26, 887-905.

[140]. Kusiak, A. [1990]. *“A branch and bound algorithm for solving the group technology problem”.* Annals of Operations Research, 26, 415-431.

[141]. Kusiak, A. [1992]. *“Group Technology: Models and solutions approaches”.* Proceedings of 1st Industrial Engineering Research Conference, 349-352.

[142]. Lawrence, S., [1984]. *“Supplement to resource constrained project scheduling: An experimental investigation of heuristic scheduling*

techniques". Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, P.A.

[143]. Lee, M.K., Luong, H.S. and Abhary, K., [1997]. *"A genetic algorithm based cell design considering alternative routing"*. Computer Integrated Manufacturing Systems, 10, 93–107.

[144]. Leem, C.W. and Chen, J.J.G., [1996]. *"Fuzzy-set-based machine-cell formation in cellular manufacturing"*. International Journal of Manufacturing, 7, 355–364.

[145]. Lee-Post, A., [2000]. *"Part family identification using a simple genetic algorithm"*. International Journal of Production Research, 38, 793–810.

[146]. Lin, Shih-Wei, Gupta, Jatinder N. D., Ying, Kuo-Ching and Lee, Zne-Jung [2008]. *"Using simulated annealing to schedule a flowshop manufacturing cell with sequence-dependent family setup times"*, International Journal of Production Research, 1-13.

[147]. Levuhis, R., [1978]. *"Group Technology--A Review of the State of the Art in the United States"*. Chicago: K.W. Tunnel Co.

[148]. Li, J., Chu, C. -H., Wang, Y. and Yan, W., [2007]. *"An improved fuzzy clustering method for cellular manufacturing"*. International Journal of Production Research, 45(5), 1049-1062.

[149]. Liao, T. W. [2001]. *"Classification and Coding approaches to part family formation under a fuzzy environment"*. Fuzzy Sets and Systems, 122(3), 425-441.

[150]. Liaw C. F, [2000]. *"A Hybrid Genetic Algorithm for the Open Shop Scheduling Problem"*. European Journal of Operation Research 124, 28-42.

[151]. Logendran, R., [1990]. *"A workload based model for minimizing total intercell and intracell moves in cellular manufacturing"*. International Journal of Production Research, 28(5), 913-925.

[152]. Love, D.M. and George, A.R. [1985]. "*Designing a computerised component coding and classification system to minimise implementation costs"*. Proceedings CADCAM'85, Cambridge, UK, 23–31.

[153]. Lozano, S., Adenso-Diaz, B., Eguia, I., Onieva, L., [1999]. *"A one-step tabu search algorithm for manufacturing cell design"*. Journal of the Operational Research Society, 50, 509–516.

[154]. Lundy, M., and Mees, A., [1986]. *"Convergence of an annealing algorithm. Mathematical Programming"*. 34, 111-124.

[155]. Malave, C. O., Ramchandran, S., [1991]. *"A neural network based design of Cellular Manufacturing System"*. Journal of Intelligent Manufacturing, 2, 305-314.

[156]. Manne A. S., [1960]. *"On the Job Shop Scheduling Problem"*. Operation Research, 8, 219-223.

[157]. Martin P. D., [1991]. *"A Time Oriented Approach to Computing Optimal Schedules for Job Shop Scheduling Problem"*. PhD Thesis, School of Operation Research & Industrial Engineering, Cornell University, New York.

[158]. Masaru, T., Hiji, M., Miyabayashi, K., Okumura, K., [2000]. *"A New Genetic Representation and Common Cluster Crossover for Job Shop Scheduling Problems"*. Evo. Workshops, 297-306.

[159]. Meeran S., [2003]. *"A History of Encounters with Intelligent Search Job Shop Scheduling-Failures, Successes and Lessons Learnt"*. Proceedings of 19th International Conference on CAD/CAM Robotics and Factories of the Future, Kuala Lumpur, Malaysia, 22-24 July, K3-1-K3-23.

[160]. McAuley, J., [1972]. *"Machine grouping for efficient production"*. Production Engineer, 51(2), 53-57.

[161]. McCarthy J., [1960]. *"Recursive Functions of Symbolic Expressions and their Computation by Machine"*. Part 1, Communications of the ACM, 3, 475-482.

[162]. McCormick, W. T., Schweitzer, P. J., White, T. W. [1972]. *"Problem decomposition and data reorganization by a clustering technique"*. Operations Research, 20, 993-1009.

[163]. Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N. & Teller, A. H., [1953]. *"Equation of state calculation by fast computing machines"*. Journal of Chemical Physics, 21, 1087-1092.

[164]. Mitrofanov, S. P., [1966]. *"Scientific Principles of Group Technology"*. Yorkshire, UK.

[165]. Mohan G. B. and Florian M., [1975]. *"On Scheduling with Ready Times and Due Dates to Minimize Maximum Lateness"*. Operation Research, 23(3), 475 – 482.

[166]. Momim M., [1999]. *"A knowledge-Based Approach for the Dynamic Scheduling and Sequencing in Manufacturing Cells"*. PhD Thesis, University of Bradford, UK.

[167]. Montana D. J., [1992]. *"Neural Network Weight Selection Using Genetic Algorithm"*. Bolt Beranek and Newman Inc. 70 Fawcett Street, Cambridge, MA02138.

[168]. Moon, Y. B., [1990]. *"Forming part families for cellular manufacturing: A neural network approach"*. International Journal of Advanced Manufacturing Technology, 5, 278-291.

[169]. Moon, Y. B., [1991]. *"An interactive and competition for machine part family formation in group technology"*. Proceedings of International Joint Conference Neural Networks, Washington, USA, 667-670.

[170]. Moon, Y. B., & Chi, S. C., [1992]. *"Generalized part family formation using neural network techniques"*. JMS, 11(3), 149-159.

[171]. Moon, C. and Gen, M., [1999]. *"A genetic algorithm-based approach for design of independent manufacturing cells"*. International Journal of Production Economics, 60/61, 421–426.

[172]. Moon, C. and Kim, J., [1999]. *"Genetic algorithm for maximizing the parts flow within cells in manufacturing cell design"*. Computers & Industrial Engineering, 36, 379–389.

[173]. Moon, Y. B., Roy, U., [1992]. *"Learning group technology pat families from solid models from parallel distributed processing"*. International Jounal of Advanced Manufacturing Technology, 7, 109-118.

[174]. Morshed M. S., [2006]. *"A Hybrid Model for Job Shop Scheduling"*. PhD Thesis, University of Birmingham.

[175]. Mosier, C.T. & Taube, L. [1985]. *"A weighted similarity coefficient for use in addressing the group technology part machine grouping problem"*. American Institute of Decision sciences Annual Meeting, Los Vegas, 812-815.

[176]. Mosier, C.T. & Taube, L. [1985a]. *"The facets of group technology and their impact on implementation"*. OMEGA, 13(6), 381-391.

[177]. Mosier, C.T. & Taube, L. [1985b]. *"Weighted similarity measure heuristics for the group technology machine clustering problem"*. OMEGA, 13(6), 577-583.

[178]. Mungwattana, A., [2000]. *"Design of Cellular Manufacturing Systems for dynamic and uncertain production requirements with presence of routing flexibility"*. PhD thesis, Virginia Polytechnic Institute and State University, Blacksburg, VA.

[179]. Murugan, M., & Selladurai, [2005]. *"Manufacturing cell design with reduction in setup time through genetic algorithm".* Journal of Theoretical and Applied Information Technology, 76-97.

[180]. Nakano, R., Yamada, T., [1991]. *"Conventional genetic algorithms for Job-Shop problem".* Proceedings of the 4th International Conference on Genetic Algorithms and their Applications, San Diego, California, USA, 474-479.

[181]. Negnevitsky M., [2002]. *"Artificial Intelligence--A guide to intelligent Systems".* Addison-Wesley, Essex CM20 2JE, England. 1st Edition.

[182]. Ng, S. [1992]. *"On the measures of cell formation in group technology".* Proceedings of the 1st Industrial Engineering Conference, 353-356.

[183]. Ng, S. [1993]. *"Worst-case analysis of an algorithm for cellular manufacturing".* EJOR, 69(3), 384-398.

[184]. Nemhauser G. L. & Wolsey L. A. (Eds), [1988]. *"Integer and Combinatorial Optimization".* John Wiley and Sons, New York.

[185]. Noor, S., [2007]. *"Operational scheduling of traditional and flexible manufacturing systems using genetic algorithms, artificial neural networks and simulation".* PhD Thesis, University of Bradford, UK.

[186]. Noor S., Khan M. K., Hussain I., Ullah I., [2005]. *"Application of Simulation and Artificial Neural Network to the problem of Scheduling of Flexible Manufacturing System".* International Journal of INGENIUM, 191-199.

[187]. Noor S, Khan M. K., Hussain I, Ullah I., [2006]. *"Scheduling Tool for the Flexible Manufacturing Systems using Hybrid Genetic Algorithm".* 2nd International Conference on Emerging Technologies, Peshawar, Pakistan.

[188]. Norman, B., Bean, J., [1995]. *"Random keys genetic algorithm for Job-Shop scheduling: Unabridged version".* Technical Report, Department of Industrial and Operations Engineering, University of Michigan, Ann Arbor, USA.

[189]. Nsakanda, A. L., Diaby, M., & Price, W. L., [2005]. *"Hybrid genetic approach for solving large-scale capacitated cell formation problems with multiple routings".* European Journal of Operations Research, (Article in press).

[190]. Obitko M. and Slavík P., [1999]. *"Visualization of Genetic Algorithms in a Learning Environment".* Spring Conference on Computer Graphics, CCG'99. Bratislava: Comenius University, ISBN 80-223-1357-2, 1999, 101-106.

[191]. Offodile, O. F., [1991]. *"Application of Similarity Coefficient Method to Parts Coding and Classification Analysis in Group Technology".* Journal of Manufacturing Systems, 10(6), 442-448.

[192]. Oliver I,, Smith D., Holland J., [1987]. *"A Study of Permutation of Crossover Operators on the Traveling Saleman Problem".* Proceedings of the Second International Conference on Genetic Algoirthms, Hillsdale, NJ:Lawrence Erlbaum Associates, 224-230.

[193]. Ono I., Yamamura M., Kobayashi S., [1996]. *"A Genetic Algorithm for Job Based Crossover".* Proceedings of the Third IEEE Conference on Evolutionary Computation, Japan, 547-552.

[194]. Onwubolu G. C., [2000]. *"Manufacturing Cell Scheduling Using Genetic Algorithms".* Proceedings of Institution of Mechanical Engineers, 214, Part B, 159-164.

[195]. Onwubolu, G.C., Mlilo, P.T., [1998]. *"Manufacturing cell grouping using similarity coefficient-distance measure".* Production Planning & Control, 9, 489–493.

[196]. Onwubolu, G.C., & Mutingi, M. [2001]. *"A genetic algorithm approach to Cellular Manufacturing Systems".* Computers and Industrial Engineering, 39(1-2), 125-144.

[197]. Opitz, H. [1970]. *"A Classification System to Describe Work Pieces".* New York: Pergamon Press.

[198]. Opitz, H., Eversheim, W., & Wiendhal, H.P. [1969]. *"Work piece Classification and Its Industrial Applications".* International Journal of Machine Tool Design and Research, 9, 39-50.

[199]. Peker, A. and Kara, Y., [2004]. *"Parameter setting of the Fuzzy ART neural network to part-machine cell formation problem".* International Journal of Production Research, 42(6), 1257–1278.

[200]. Plaquin, M., Pierreval, H., [2000]. *"Cell formation using evolutionary algorithms with certain constraints".* International Journal Of Production Economics, 64(1-3), 267-278.

[201]. Pohlheim H., [2005]. *"Genetic and Evolutionary Algorithm Toolbox for use with MATLAB"* [online] Available at: http://www.geatbx.com/docu/algindex-02.html#P452_27492

[202]. Pullen, R. D., [1976]. *"A survey of Cellular Manufacturing cells"*. The Production Engineer, 55(9), 451-454.

[203]. Purcheck, G. F. K., [1974]. *"A mathematical classification as a basis for the design of group technology production cells"*. Production Engineer, 54(1), 35-48.

[204]. Purcheck, G. F. K., [1975]. *"Combinatorial grouping - a lattice theoretic method for the design manufacturing systems"*. Journal of Cybernetics.

[205]. Rajagopalan, R., Batra, J. L., [1975]. *"Design of cellular production system: A graph theoretic approach"*. International Journal of Production Research, 13(6), 567-579.

[206]. Ribeiro, J.F.F., [2009]. "*Manufacturing cells formation based on graph theory*". International Conference on Computers and Industrial Engineering (ICCIE), 6-9 July, Troyes, France. 658-662.

[207]. Rich E. and Knight K., [1991]. *"Artificial Intelligence"*. 2nd Edition, MGraw-Hill, Inc.

[208]. Safaei, N., Saidi-Mehrabad, M., Jabal-Ameli, M.S., [2008]. *"A hybrid simulated annealing for solving an extended model of dynamic Cellular Manufacturing System"*. European Journal of Operational Research, 185(2), 563-592.

[209]. Sakawa M., Kubota R., [2000]. *"Fuzzy Programming for Multi-Objective Job Shop Scheduling with Fuzzy Processing Time and Fuzz Due Date Through Genetic Algorithms"*. European Journal of Operational Research, 120(2), 393-407.

[210]. Satake, T.,Morikawa, K., and Nakamura,N., [1994]. *"Neural network approach for minimizing the Makespan of the general Job-Shop"*. International Journal of Production Economics, 33 (1-3), 67-74.

[211]. Seifoddini, H., [1990]. *"A probabilistic model for cell formation"*. JMS, 9(1), 69-75.

[212]. Seifoddini, H. & Djassemi, M. [1995]. *"Merits of the production volume based similarity coefficient in machine cell formation"*. JMS, 14(1), 35-44.

[213]. Seifoddini, H. & Wolfe, P. [1986]. *"Application of Similarity Coefficient Method in Group Technology"*. IIE Transactions, 18(3), 271-277.

[214]. Seifoddini, H. [1989]. *"Single linkage versus average linkage clustering in machine cells formation applications"*. Computers and Industrial Engineering, 16(3), 419-426.

[215]. Selim, H. Askin, R. [1998]. *"Cell Formation in Group Technology: Review Evaluation and Directions for Future Research"*. Computers and Industrial Engineering, 34(1): 3-20.

[216]. Serdar Uckun, Bagchi, S., Kawamura, K., Miyabi, Y., [1993]. *"Managing Genetic Search in Job Shop Scheduling"*. IEEE Experts, 15-24.

[217]. Sexton R. S., Gupta J. N. D., [2000]. *"Comparative evaluation of Genetic algorithm and Backpropagation for training neural networks"*. Information Science 129, 45-49.

[218]. Shafer, S. and Charnes, J. [1994]. *"Cellular versus Functional Layouts under a Variety of Shop Operating Conditions"*. Decision Sciences, 24(3):665-682.

[219]. Shafer, S. M. & Rogers, D. F., [1993b]. *"Similarity and distance measures for cellular manufacturing: An extension and comparison"*. International Journal of Production Research, 31(6), 1315-1326.

[220]. Shazly M. R. E., Shazly H. E. E., [1999]. *"Forecasting currency prices using genetically evolved neural network architecture"*. International Review of Financial Analysis, 8(1), 67-82.

[221]. Shtub, A. [1989]. *"Modelling Group Technology Cell Formation as a Generalized Assignment Problem"*. International Journal of Production Research, 27(5), 775-782.

[222]. Shunk, D.L. [1978]. *"Computer Integrated Manufacturing" in Manufacturing High technology Hand Book,* New York, 83-100.

[223]. Singh, N. & Rajamaani, D. [1996]. *"Cellular Manufacturing Systems: Design, Planning and Control"*. Chapman & Hall, New York.

[224]. Slagle, J. L., Chang, C. L., & Heller, S. R., [1974]. *"A clustering and data reorganisation algorithm"*. IEEE Transactions on Systems, Man, and Cybernetics, 5(2), 125-128.

[225]. Smith S. B., [1989]. *"Computer Based Production and Inventory Control"*. Prentice-Hall International, Inc.

[226]. Sofianopoulou, S., [1997]. *"Application of simulated annealing to a linear model for the formulation of machine cells in group technology"*. International Journal of Production Research, 35(2), 501-511.

[227]. Souri M., [2003]. *"Dictionary of IT Terms"*. Pentagon Press, New Delhi.

[228]. Srinivasan, G. [1994]. *"A clustering algorithm for machine cell formation in group technology using minimum spanning trees"*. International Journal of Production Research, 32, 2149-2158.

[229]. Srinivasan, G., Narendran, T. & Mahadevan, B. [1990]. *"An assignment model for the part-families problem in group technology"*. International Journal of Production Research, 28(l), 145-152.

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

[231]. Stanfel, L. E. [1985]. *"Machine clustering for economic production"*. Engineering Costs and Production Economics, 9, 73-81.

[232]. Su, C.T. and Hsu, C.M., [1996]. *"A two-phased genetic algorithm for the cell formation problem"*. International Journal Industrial Engineering, 3, 114-125.

[233]. Su, C.T. & Hsu, C.M., [1998]. *"Multi-objective machine–part cell formation through parallel simulated annealing"*. International Journal of Production Research, 36, 2185-2207.

[234]. Sunderesh and Heragu, S., [1994]. *"Group Technology and Cellular Manufacturing"*. IEEE Transactions on Systems, Man, and Cybernetics, 24(2).

[235]. Suresh, N. and Meredith, J. [1994]. *"Coping with the Loss of Pooling Synergy in Cellular Manufacturing Systems"*. Management Science, 40(4), 466-483.

[236]. Suresh, N. C., & Kaparthi, S., [1994]. *"Performance of fuzzy art neural network for group technology cell formation"*. International Journal of Production Research, 32(7), 1693-1713.

[237]. Suresh, N. C., & Park, S., [2003]. *"Performance of Fuzzy ART neural network and hierarchical clustering for part-machine grouping based on operation sequences"*. International Journal of Production Research, 41(14), 3185-3216.

[238]. Susanto, S., Kennedy, R.D. and Price, J.W.H., [1999]. *"A new fuzzy-c-means and assignment technique-based cell formation algorithm to perform part-type clusters and machine type clusters separately"*. Production Planning & Control, 10(4), 375–388.

[239]. Syswerda G., [1989]. *"Uniform Crossover Genetic Algorithms"*. Proceedings of the Third International Conference on Genetic Algorithms, San Mateo, CA, Morgan Kaufmann Publishers, 2-9.

[240]. Tariq, A., Hussain, I., Ghaffoor, A., [2006]. *"A hybrid genetic algorithm for Machine-Part grouping"*. Proceedings of the 2nd International Conference on Emerging Technologies (ICET), Peshawar, Pakistan, Nov. 13-14.

[241]. Tariq, A., Hussain, I., Ghaffoor, A., [2007]. *"A hybrid genetic algorithm for Job-Shop scheduling"*. 37th Internation Conference on Computer and Industrial Engineering, Alexandria, Egypt.

[242]. Tariq, A., Hussain, I., Ghaffoor, A., [2007]. *"Consideration of single machine cells in designing Cellular Manufacturing System using a Hybrid Genetic Algorithm"*. Proceedings of the 3rd International Conference on Emerging Technologies (ICET), Islamabad, Pakistan, Nov. 12-13.

[243]. Tariq, A., Hussain, I., Ghaffoor, A., [2009]. *"A Hybrid Genetic Algorithm for Machine Part Grouping" (Extended version).* Computers and Industrial Engineering, 56(1), 347-356.

[244]. Tamaki, H., & Nishikawa, Y., [1992]. *"A parallel genetic algorithm based on neighbourhood model and its application to the Job-Shop scheduling"*. Proceedings of the 2nd International Conference on Parallel Problem Solving from Nature, Brussels, Belgium, 573-582.

[245]. Tavakkoli-Moghaddam, R., Aryanezhad, M. B., Safaei, N. Azaron, A., [2005]. *"Solving a dynamic cell formation problem using metaheuristics"*. Applied Mathematics and Computation, 170(2), 761-780

[246]. Tsakonas A., Dounias G., [2002]. *"Hybrid Computational Intelligence Schemes in Complex Domains"*. University of the Aegean [online] Available at: http//:decision.ba.aegan.gr

[247]. Uddin, M. K., Shanker, K., [2002]. *"Grouping of parts and machines in presence of alternative process routes by genetic algorithm"*. International Journal of Production Economics, 76(3) 219-228.

[248]. Vannelli, A., & Kumar, K. R., [1986]. *"A method for finding minimal bottleneck cells for grouping part-machine families"*. International Journal of Production Research, 24(2), 387-400.

[249]. Venkataramanaiah, S. [2007]. *"Scheduling in Cellular Manufacturing Systems: an heuristic approach"*. InternationalJournal of Production Research, 46(2), 429-449.

[250]. Venugopal, V., Narendaran, T. T., [1992a]. *"Cell formation in manufacturing systems through simulated annealing: An experimental evaluation"*. EJOR, 63(3), 409-422.

[251]. Venugopal, V., Narendaran, T. T., [1992b]. *"A genetic algorithm approach to the machine component grouping problem with multiple objectives"*. Computers & Industrial Engineering 22(4), 469-480.

[252]. Veilleux R. F. and Petro L. W., [1988]. *"Tool and Manufacturing Engineers Handbook Vol 5, Manufacturing Management*". Society of Manufacturing Engineers (SME), One SME Drive Dearbon, Michingan, 1988.

[253]. Vohra, T., Cheng, D., Chang, J., & Chen, H., [1990]. *"A network approach to cell formation in cellular manufacturing"*. International Journal of Production Research, 28(11), 2075-2084.

[254]. Vollmann T. E., Berry W. L. and Whybark D. C., [1997]. *"Manufacturing Planning and Control Systems"*. Homewood, Illinois Irwin.

[255]. Vondermbse M.A. and White G.P., [1991]. *"Operation Management: Concepts, Methods and Strategies"*. West Publishing Company.

[256]. Waghodekar, P. H., Sahu, S. [1984]. *"Machine-component cell formation in group technology: MACE"*. International Journal of Production Research, 22(6), 937-948.

[257]. Wang L. and Zheng D., [2001]. *"An effective hybrid optimization strategy for Job-Shop scheduling problems"*. Computers and Operations Research 28, 585-596.

[258]. Wemmerlov, U., & N. L. Hyer [1989]. *"Cellular Manufacturing practices"*. Manufacturing Engineering, 102(3), 79-82.

[259]. Wemmerlov, U. [1984]. *"Comments on direct clustering algorithm for group formation in cellular manufacturing"*. JMS, 3, vii-ix.

[260]. Wei, J. C., & Gaither, N., [1990]. *"An optimal model for cell formation decisions"*. Decision Sciences, 21(2), 416-433.

[261]. Wei, J. C., Kern, G. M., [1989]. *"Commonality analysis: A linear cell clustering algorithm for group technology"*. International Journal of Production Research, 27(12), 2053-2062.

[262]. White, J. D. E., [1980]. *"The use of similarity coefficient in production flow analysis"*. International Journal of Production Research, 18(4), 504-514.

[263]. Wiers V. C. S., [1997]. *"A review of the Applicability of OR and AI Scheduling Techniques in Practice"*. Omega, International Journal of Management Science, 25(2).

[264]. Wilhelm, M. R., & Ward, T. L., [1987]. *"Solving quadratic assignment problems by `simulated annealing"*. IIE Transactions, 19, 107-119.

[265]. Won, Y., [2000]. *"Two-phase approach to GT cell formation using efficient p-median formulations"*. International Journal of Production Research, 38(7) 1601- 1613.

[266]. Won, Y. and Currie, K. R., [2007]. *"Fuzzy ART/RRR-RSS: a two-phase neural network algorithm for part-machine grouping in cellular manufacturing"*, International Journal of Production Research, 45(9), 2073-2104.

[267]. Wu, D. [1987]. *"An Expert Systems Approach for the Control and Scheduling of Flexible Manufacturing Systems"*. Ph.D. Dissertation, Pennsylvania State University.

[268]. Wu, X.D., Chu, C.H., Wang, Y.F. and Yan, W.L., [2002]. *"A genetic algorithm for integrated cell formation and layout decision"*. Proceedings of IEEE Congress on Evolutionary Computation, Honolulu, Hawaii, 12–17 May.

[269]. Wu, X., Chu, C. -H., Wang, Y. and Yan, W., [2006]. *"Concurrent design of Cellular Manufacturing Systems: a genetic algorithm approach"*. International Journal of Production Research, 44(6), 1217-1241

[270]. Xambre, A. R., Vilarinho, P. M., [2003]. *"A simulated annealing approach for manufacturing cell formation with multiple identical machines"*. European Journal of Operational Research, 151(2), 434-446.

[271]. Xu, H., & Wang, H. P., [1989]. *"Part family formation for GT application based on fuzzy mathematics"*. International Journal of Production Research, 27(9), 1637-1651.

[272]. Yamada, T., Nakano, R., [1992]. *"A genetic algorithm applicable to job shop problems"*. Proceedings of the 2nd International Workshop on Parallel Problem Solving from Nature, Brussels, Belgium, 281-290.

[273]. Yasuda, K., Yin, Y., [2001]. *"A dissimilarity measure for solving the cell formation problem in cellular manufacturing"*. Computers and Industrial Engineering, 39, 1–17.

[274]. Yasuda, K., Hu, L., Yin, Y., [2005]. *"A grouping genetic algorithm for multi-objective cell formation problem"*. International Journal of Production Research, 43(4), 829–853.

[275]. Yeun Y.S., Lee K.H., Yang Y.S., [1999]. *"Function approximation by coupling neural networks and genetic programming trees with oblique decision trees"*. Artificial Intelligence in Eng. 13, 223-239.

[276]. Yin, Y., Yasuda, K., [2006]. *"Similarity coefficient methods applied to the cell formation problem: A taxonomy and review"*. International Journal of Production Economics, 101, 329-352.

[277]. Zadeh, L. A., [1965]. *"Fuzzy sets"*. Information and Control, 8, 338-353.

[278]. Zhang, H. C., and Huang, S.H., [1995]. *"Applications of neural networks in manufacturing: a state-of-the-art survey"*. International Journal of Production Research, 33(3), 705-728.

[279]. Zhang, C., Wang, H., [1992]. *"Concurrent formation of part families and machine cells based on the fuzzy set theory"*. JMS, 11(1), 61-67.

[280]. Zhao, C., & Wu, Z. [2000]. *"A genetic algorithm for manufacturing cell formation with multiple routes and multiple objectives"*. International Journal of Production Research, 38(2), 385-395.

[281]. Zhou, D. N., Cherkassy, V., Baldwin, T. R., and Olson, D. E., [1991]. *"A neural network approach to Job-Shop scheduling"*. IEEE Transactions on Neural Networks, 2(1), 175-179.

[282]. Zhou H., Feng Y. and Han L., [2001]. *"The Hybrid Genetic Algorithm for Job Shop Scheduling"*. Computers and Industrial Engineering, 40, 191-200.