Founded 1905

PLANNING IN PARALLEL MACHINE SHOPS AND SCHEDULING OF FLEXIBLE PROCESS PLANS IN MOULD MANUFACTURING SHOP

BY

SARAVANAKUMAR MOHANRAJ

(B.E., M.E.)

A THESIS SUBMITTED

FOR THE DEGREE OF MASTER OF ENGINEERING

DEPARTMENT OF MECHANICAL ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2004

ACKNOWLEDGEMENTS

I am very happy to take this opportunity to thank Dr. Lee Kim Seng for his invaluable guidance, thought sharing and suggestions throughout my research period at the National University of Singapore. His ideas and recommendations have played a significant role in completing this work successfully. Further, I extend my sincere and deepest gratitude for his priceless advice, motivation and moral support throughout my stay here in Singapore.

I would also like to thank all the faculty members in Department of Mechanical Engineering, National University of Singapore for their professional advice in enlightening me on issues pertaining to my research. I am grateful for their critical suggestions in this project.

I would like to thank all my friends and colleagues who helped me in one way or other to carry out the research work successfully. In particular, I wish to thank Dr Sun Yifeng, Dr. Mohammad Rabiul Alam, Mr Woon Yong Khai, Miss Du Xiaojun, Miss Maria Low Leng Hwa, Ms Cao Jian, Mr Atiqur Rahman, and Miss Zhu Yada for actively participating in the discussion related to my project work and creating a wonderful environment that made my study in NUS an enjoyable and memorable one.

I wish to thank my parents (Mr. M.Mohanraj and Mrs. M.Banumathi) and my sister (Miss M. Jamuna Rani) for their moral support.

I am thankful to the National University of Singapore for providing me a chance to pursue my research work and financing with research scholarship to support my studies.

TABLE OF CONTENTS

SUMMARY

Sequencing and scheduling considerations prevalent in multiple identical processors with constraints have been addressed in this work. Heuristic techniques are necessary and sometimes only hope to study the critical parameters in single stage or entire structure of the complex manufacturing systems. In this research, two heuristic algorithms are developed to study the critical parameters of sequencing and scheduling tasks in parallel machine shops. One of the developed heuristic algorithms provides the polynomial time solution for scheduling problems in identical parallel machine shop environment. In order to explore the planning problems in large scale systems, Meta heuristic techniques are studied. Necessities of Meta heuristic techniques are becoming essential to obtain the better solution for non linear optimization problems. Some of the sequencing and scheduling problems in parallel machine shops are proved to be NP-Hard (Non-deterministic Polynomial) problems. Finding the optimum solution using conventional optimization techniques will take large amount of time. Even with long computational time, there is no guarantee for optimum solution with conventional techniques. In this research, four Meta heuristic techniques namely genetic algorithm, simulated annealing algorithm, memetic algorithm and tabu search are modified for the suitability of parallel machine shop environment and simulated for various measures in order to achieve better solution. The performance of the Meta heuristic techniques are compared by DOE (Design of Experiment) technique. A new approach is also developed to combine the sequencing and scheduling tasks in parallel machine shops. Simulation is carried out to test the effects of the proposed approach in parallel machine shop environment. Scheduling and process planning used to be two very separate processes in most of the manufacturing shops. However, due to the recognition of the intricate relationship between them, a number of researches have recently focused on integrating these two

processes. The usage of flexible process plans in scheduling task allows more flexibility in production control and results in substantial cost savings. However, it also increases the solution space of the optimization problem and makes it more critical to have an effective optimization algorithm than traditional techniques.

This thesis also deals with scheduling of flexible process plans in mould manufacturing system. Two Meta heuristic algorithms namely, genetic algorithm and simulated annealing approaches are used to solve the mould shop scheduling problems. Various performance measures are considered while evaluating the system, such as makespan, total flow time, total lateness and combined objective function. Instead of considering a flexible manufacturing system like many other researches, this project focuses on the demands of semi automated factories, which are especially true for mould manufacturing shops. The project also involves in deciding the optimization algorithm, the methodology of the algorithm and effectiveness of performance measure for mould manufacturing shop. Additional attention is paid to study the adoptability of Meta heuristic techniques in mould manufacturing shop. The contribution of this research includes the development of heuristic approaches, accessing the suitability of Meta heuristic methods in sequencing and scheduling tasks of parallel machine shops and developing the new approach to solve the planning tasks concurrently in parallel machine clusters. A new approach is also proposed to schedule the flexible process plans in mould manufacturing shop. Two Meta heuristic techniques are used to evaluate the new approach. Parameters of the algorithms are modified in order to suit the mould manufacturing environment. Modified Meta heuristic techniques produce better schedules which can help to improve the planning tasks in mould shop.

NOMENCLATURE

t_{ijk}	Processing time of job i, operation j on machine k
d_i	Due date of job i
P_i	Priority of job i
s_{jik}	Starting time of job i, operation j on machine k
c_{jik}	Completion time of job i, operation j on machine k
C_i	Completion time of all the operations in job i
S_i	Slack time available in job i
α	Penalty factor assigned for jobs finishing tardy
β	Penalty factor assigned for jobs finishing early
F(S)	Objective function based on the schedule S
EL_i	Earliness of job i
TL_i	Lateness of job i
TR_i	Tardiness of job i
FL_i	Flow time of job i
C_{max}	Makespan of a schedule
L_{max}	Maximum Lateness
m	Total number of machines in the system
n	Total number of jobs available for scheduling
NP	Non-deterministic Polynomial
Nc	Total number of clusters
MNc_x	Number of machines available in cluster x
FMS	Flexible Manufacturing System
FMC	Flexible Manufacturing Cell
CIM	Computer Integrated Manufacturing

CAPP Computer Aided Process Planning

CAD Computer Aided Design

CAM Computer Aided Manufacturing

WFL_i Priority weighted flow time of job i

WTR_i Priority weighted tardiness of job i

WEL_i Priority weighted earliness of job i

WTL_i Priority weighted total lateness of job i

Comobject Combined measure of multiple objectives with weights

S_c Current schedule

S_k Alternative schedule

pop_size Population size

p_c Crossover Probability

p_m Mutation Probability

max_gen Maximum number of generation cycles

T Temperature

T_θ Lowest temperature value

T_0 Initial temperature value

$\{SS\}_w$ Sequenced set of jobs in ascending order of processing time

$\{SS\}_e$ Sequenced set of jobs in descending order of processing time

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

BACKGROUND

In today's competitive market, integration of various manufacturing activities is very important to obtain the better lead time. Lead time of the product became an important criterion, which decides the performance of manufacturing industry. The competition for quicker product release among manufacturing industries leads to the development of automated systems such as Flexible Manufacturing Systems (FMS), Flexible Manufacturing Cells (FMC), Computer Integrated Manufacturing (CIM) and etc. However implementation of all these approaches is not possible to all the manufacturing industries. In order to achieve the better lead time of the product higher attention has to be given on value addition activities, such as machining. Even though the machining time of the product cannot be reduced without much technological advancement, there are chances of reducing the planning times associated to machining of the product by considering the flexibilities in manufacturing environment.

Scheduling and process planning are two of the main planning activities which can be improved in manufacturing shops. Flexibilities exist in planning activities in the forms of number of identical machines to process the same operation, alternatives of operations to produce the same part and etc. These flexibilities can be used to reduce the overall planning time of the manufacturing activity. In addition to flexibilities, there are constraints which have to be satisfied in order to achieve the feasible solution in any manufacturing system. However, suitable approaches are necessary to improve the scheduling and process planning tasks in manufacturing shops.

1.1 META HEURISTICS AND SEQUENCING & SCHEDULING

1.1.1 Introduction to Meta Heuristics

A heuristic approach to a problem is an empirical search or near optimization method that often works at solving the problem, but does not have any rigorous proof that people like physicists and mathematicians expect. It is also referred to as “a rule” that provides a shortcut to solve difficult problems. Heuristics are used when there is limited time and/or information to make a decision. In general, heuristics are formed by “rule of thumb”. Unlike algorithms, heuristics do not guarantee optimal and are often used with no theoretical guarantee. Heuristics are efficient while solving or studying the prototype system or stage in the complex system. However for large scale systems, there is a need for better technique which can combine heuristic algorithms.

A Meta Heuristic is a semi-mythical method for finding good heuristics and is used in solving particular problems. It is further explained as a collection of heuristic algorithms applicable to a wide variety of problems. As mentioned before, the heuristic approach is often associated with “rules of thumb” or clever insights. Based on the heuristics provided, the Meta heuristic algorithm performs the search process iteratively to look for a solution. The iterative search process is terminated when no improvements are possible. The choice of meta-heuristic algorithm depends on parameters, such as the solution quality required and the availability of problem knowledge. Some of the popular Meta heuristic methods are thoroughly studied and explained in the subsequent chapters.

1.1.2 Introduction to Sequencing and Scheduling

The practical problem of allocating resources over time to perform a collection of tasks arises in a variety of situations. By definition, scheduling is defined as allocation of jobs to machines and sequencing is the arrangement of jobs to the allocated machines. These

two functions can be performed either individually or simultaneously. Scheduling theory and approach will vary based on the system structure such as single machine scheduling, parallel machine scheduling and job shop scheduling. In single machine systems, the pure sequencing problem is a specialized scheduling problem in which an ordering of jobs completely determines a schedule. In typical parallel machine systems, jobs are considered to be scheduled in any one of the available identical parallel machines. In some cases the performances of the machines are also included in the specification of parallel machine system. In addition to single machine and parallel machine systems, job shop and flow shops are other important scheduling environment. One important difference in a typical job shop system from other scheduling systems is the flow of work is not unidirectional in a job shop. The job shop scheduling problem is one of the most complex machine scheduling problems. The criterion called “Routing” in job shop scheduling decides the flow of jobs in the system. Routing also gives the precedence constraints between the operations in each job. Complexity of the job shop scheduling problem is explained with a simple example.

Table 1.1 Processing time matrix

Operation / Job	1	2	3
1	4	3	2
2	1	4	4
3	3	2	3
4	3	3	1

Consider 4 jobs and 3 machines in a job shop environment with each job consisting of 3 operations. Processing time of each operation in the jobs is given in Table 1.1 and Routing is given in Table 1.2.

Table 1.2 Routing matrix

Operation / Job	1	2	3
1	1	2	3
2	2	1	3
3	3	2	1
4	2	3	1

For a general job shop problem with n jobs and m machines contains $(n!)^m$ schedules. Therefore, the example problem which is considered above contains $(4!)^3$ = 13,824 schedules. One of feasible schedule for this job shop problem is given Figure 1.1.

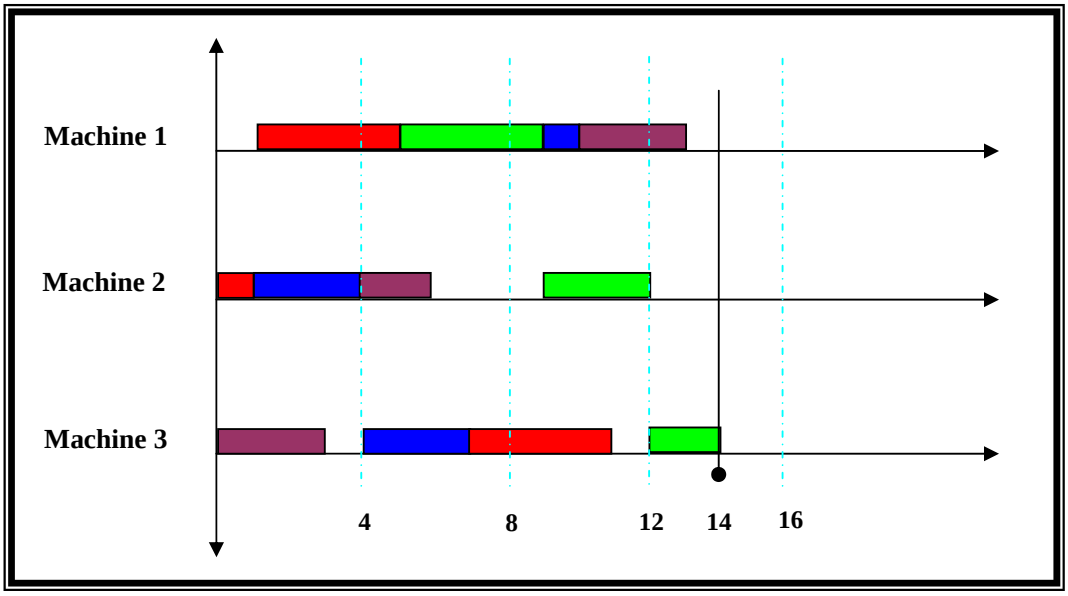

Figure 1.1 Gantt chart representing the feasible schedule for job shop

Increase in number of jobs and machines makes the solution space as hard to search for each permutation. A job shop problem and precedent constraint parallel machine scheduling problems are good examples of NP-Hard (Non deterministic Polynomial) problem as it is hard to obtain polynomial time solutions even for simple problems.

1.2 INTRODUCTION TO PROCESS PLANNING

Process planning is one of the important tasks in manufacturing which gives the detailed list of operational instructions to manufacture a product from a piece of raw material. A process plan provides lots of information which are necessary to manufacture a product such as operations, machines, tools and machining parameters. In fully automated manufacturing systems, it acts as an important tool which integrates Computer Aided Design (CAD) and Computer Aided Manufacturing (CAM). Flexibilities exist in process plans in the forms of alternative machines to produce the same part, number of identical machines in the system and etc. These flexibilities increase the chance of improving the process planning system. Along with flexibilities, there are constraints which have to be satisfied in order to obtain the feasible plan. Constraints of the process plan can be clearly explained with roughing and finishing processes, in order to achieve the feasible plan roughing has to precede the finishing process in process plan.

Even with these possibilities of improvement, still the process planning task is performed manually in most of the manufacturing industries. Mostly, suitable algorithm to make logical decisions, usage of expertise rules and searching techniques restricts the process planning to become computerized. Most of the developed computer aided process planning systems assumes unlimited resources and ideal shop floor, which are very hard to find in real manufacturing industries. Many computer aided process planning systems were reported to be limiting, time consuming, and difficult to integrate with other

planning tasks. In order to improve the planning tasks, much focus has to be given on generating the flexible process plans. With these flexibilities, there is chance of integrating and improving the planning activities in manufacturing shops. As like in mould manufacturing shop which contains several clusters of machines, grouped according to machine types. Once a job is initiated from design department, it passes through the process plan department where process plan are generated to manufacture the part. The process plan details the sequence and flexibilities to machining department that a part has to go through in order to complete all its operations.

1.3 PLASTIC INJECTION MOULD MANUFACTURING

Plastic products became an unavoidable commodity in daily life, which drives the industries for mass production of plastic products. Injection moulding is one of the main processes which help for this mass production. Thermoplastic is one of the commonly used materials in injection moulding. The schematic view of the injection moulding machine is given in Figure 1.2.

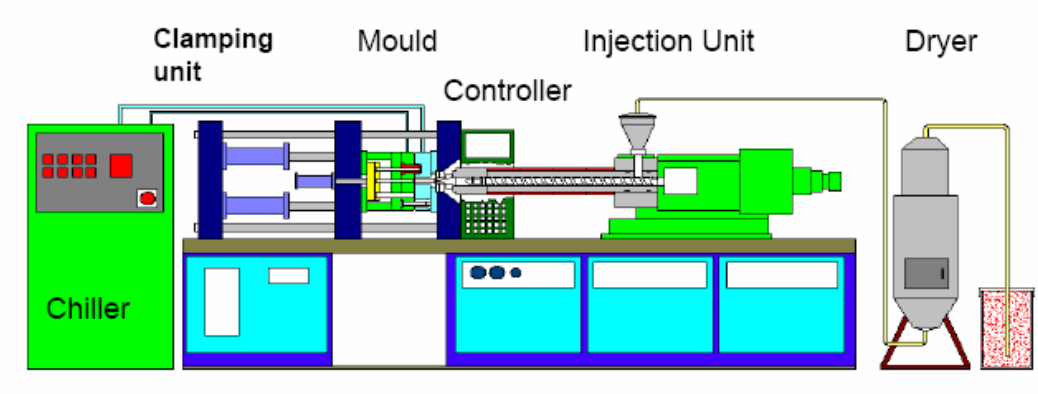

Figure 1.2 Injection moulding machine

While moulding process, the material is heated until it melts, the melted material is forced into the mould which converts the molten material into the plastic product. An

injection mould consists of core, cavity, sliders and lifters which are fitted into a mould base. The mould base is mounted on an injection moulding machine. Even this complex mechanical assembly is customized for some of the products, while injection machine can be reused for most of the products. However, dimension of the mould base and amount of power delivery during injection process limits the injection machine.

1.4 RESEARCH OBJECTIVES

Based on the above given descriptions, this research is focused on the planning problems in parallel machine shops and mould manufacturing shop. Thus, the objective for this research project is to find the suitable optimization approach for parallel machine shops and developing the system based on the chosen optimization technique to schedule the flexible process plans in mould manufacturing shop. From here, the research objective has been classified into six topics:

a) Development of the scheduling model for single stage parallel machine system and improve the scheduling function of the single stage system using the developed model and verify the suitability of the approach for multi stage production system.

b) Development of the planning model which can combine the sequencing and scheduling tasks in parallel machine system. Compare the sequential and concurrent approach for the possibility of improvement.

c) Present a scheduling model which is able to utilize flexible process plans in multistage system and also suitable to the processes involved in mould manufacturing shop.

d) Propose a Meta heuristic algorithm to effectively schedule the flexible process plans. This algorithm must be capable of considering flexible process plans while simultaneously maintaining the precedence relations between the jobs and operations.

e) Test the developed system with all the suitable performance measures to quantify the quality of solutions. The quality of the solution must reflect the need in manufacturing industry, such as in mould manufacturing shop, high priority moulds have to be finished earlier and flow time of the jobs should be minimum, thus the mould testing and modifications may have more time.

f) Test the approaches with industrial data in order to evaluate the suitability of the approaches in large scale system.

1.5 THESIS ORGANISATION

The thesis is organized as follows:

In Chapter 1, the background of Meta heuristics, sequencing and scheduling, process planning systems and the problems involved in planning tasks are identified. An introduction to the processes and equipment of plastic injection moulding is also presented.

In Chapter 2, a review of the related research in Meta heuristics and sequencing and scheduling with process planning is discussed. A detailed survey indicating the need for integration in planning tasks and optimization trend is also covered. The significance and direction of this research is clearly presented in this chapter.

Chapter 3 deals with development of heuristic algorithms for scheduling identical parallel machine cluster. The usage of heuristic algorithms in manufacturing industries is also explained in this chapter.

Chapter 4 describes the selection of Meta heuristic algorithm for scheduling task in parallel machines system and performance comparison of four Meta heuristic algorithms. This chapter also discusses the chances of applying the combinatorial optimization techniques in manufacturing shops.

Chapter 5 deals with the new approach which concurrently executes the sequencing and scheduling tasks in parallel machine system. In this chapter, two Meta heuristic algorithms are modified for the suitability of the parallel machine manufacturing systems. The importance of sequencing and scheduling are also stressed in this chapter.

Chapter 6 presents the approach for scheduling the flexible process plans in mould manufacturing shop. The solution space and complexities of this scheduling system are also explained in this chapter. The ability of the proposed approaches to schedule the mould shop planning is also explained with prototype model.

Chapter 7 presents the testing of the developed system with industrial data in order to study the performance of the system in large scale environment.

Finally, the conclusions of this project and suggestions for future works are presented.

CHAPTER 2

LITERATURE REVIEW

2.1 SEQUENCING AND SCHEDULING

Scheduling is the allocation of resources over time to perform a collection of tasks. Scheduling is mainly described as an improvement process by which limited resources are allocated over time among parallel and sequential activities. Such situations develop routinely in factories, publishing houses, shipping, universities, etc. Sequencing is the ordering of the jobs in respective machine so that there is little idle time exists between them. The pure sequencing problem is a specialized scheduling problem in which an ordering of the jobs completely determines a schedule. Moreover, the simplest pure sequencing problem is one in which there is a single resource, or machine. In manufacturing industries, this situation is generally termed as machine scheduling. The machine scheduling problems can be partitioned into two types according to the characteristics of jobs and operations. In single stage production systems, each job consists of only one operation. A single stage production system was initially referred as the basic single machine problem. The extension of the single machine problem is a parallel machine problem. The multi stage production system consists of flowshop, job shop and open shop. The group of single stage systems is also considered as multi stage systems. Most of the large scale scheduling problems is proved to be NP-hard in terms of the solution time. Two forms of NP problems exist in scheduling problem. One is NP hard and the other is NP- Complete. In terms of solution time, NP-Complete problems are easier to solve than NP-hard problems. Most of the single machine scheduling and Flow shop scheduling problems will come under NP-Complete problems. Mostly the job

shop, group of parallel machine shop and FMC scheduling problems will come under NP-hard problems.

Both the single stage parallel machine shops and multi stage parallel machine shops are most common and very important in today's industries to run the plant smoothly. There are plenty of researches that are being carried out to sequence and schedule the jobs in parallel machine system. However, till now there is no efficient model or technique to sequence and schedule the jobs in parallel machine shops in polynomial time. Another most popular scheduling problem exists in job shop environment, which serves as a yardstick for the different optimization techniques. A job shop problem starts by considering the n jobs by m machines, where each job is required to be machined by a series of machines in a particular operation sequence. The objective is then to arrange the jobs onto their machines such that none of the operation sequences are violated while achieving optimality in a performance measure. Over the years, many researches have been involved in job shop scheduling.

Flexible Manufacturing Shop (FMS) and Flexible Manufacturing Cell (FMC) contain some of the complex scheduling problems. Unlike the parallel machine systems, FMS contains groups of different types of machines in different machine departments (Hutchinson et al, 1994). In Flexible Manufacturing Cell, each cell is occupied with group of machines which are capable to complete certain job types. Both of these manufacturing concepts contain their own problems in the forms of inclusions of machines to different department in FMC and grouping of machines in FMS. Most of the FMS and FMC systems are still using the priority dispatching rules or simple heuristic technique to solve the scheduling problems (Chen and Li, 1999). Jawahar et al (1998) used the genetic algorithm approach to schedule the setup constrained FMC. Zhou and

Egbelu (1989) introduced the scheduling of machine shop systems with sequence dependent setup times.

2.1.1 Complexity in Machine Scheduling

The important concept in scheduling research is the complexity theory to classify scheduling problems as polynomially (P) solvable or NP hard (Non deterministic Polynomial). P Class consists of problems for which the execution time of the solution algorithms grows polynomial with the size of the problem. Thus, a problem of size m would be solvable in time proportional to m^k, when k is the exponent. The time taken to solve a problem belonging to the NP class grows exponentially, thus time would grow in proportion to t^m, when t is some constant. In practice, algorithms for which the execution time grows polynomially are preferred. The NP- hardness of a problem suggests that it is unlikely to find an optimal solution without the use of an essential enumerative algorithm, for which computation times will increase exponentially with problem size (Du and Leung, 1990). To obtain the exact solutions of NP-hard scheduling problems, a branch and bound, dynamic programming, or integer programming is usually applied. In most cases, these algorithms have been successful in solving problems of reasonable size, but these algorithms are restricted by the problem size. There are, however, some classes of problems that have resisted attempts to design a satisfactory solution procedure: enumerative algorithms may be unable to solve problems with more than a handful of jobs, and the solutions generated by simple methods may be far from the optimum. Recently, such problems can be efficiently tackled by Meta heuristic methods such as simulated annealing, genetic algorithm, tabu search and memetic algorithm.

In practice, it is acceptable and advisable to use Meta heuristic methods to find an approximate solution for NP-hard problem. The performance of Meta heuristic methods

is often evaluated either by computation time to reach optimal solution or analyzing the average percentage deviation from the lower bound or upper bound of that problem.

2.1.2 Approaches for solving scheduling problems

All the scheduling problem formulations engage some economic performance criteria such as makespan, flow time and lateness. It is therefore natural that these problems be optimized. Many scheduling formulations seek combinatorial solutions. However, the number of feasible schedules in combinatorial optimization grows rapidly with the number of jobs, number of machines, number of processing step, etc., making almost all problems of practical significance difficult to solve optimally. For instance, one may consider the simplest scheduling problem with n jobs to be performed in one pass manner on a single machine, with no additional constraints, when the objective is the minimization of the tardiness of the jobs from their due dates. So, there are n! possible schedules available in this case. Still most of manufacturing industries follows manual scheduling. In manual scheduling, dispatching rules are predominately used to solve scheduling problems. Dispatching rules are quite good when the production flow is smooth. Lee (1998) claimed that a perfectly designed dispatching rule with exact representation of manufacturing environment is able to provide better solutions. Even though some of the dispatching rules are efficient in solving certain scheduling problems, it is not advisable to use the dispatching rules in every environment.

Search algorithms are another popular method which has gained much attention in recent years to solve the scheduling problems. Naumann and Gu (1997) used the fuzzy logic rules to solve the scheduling problems in Flexible Manufacturing Shops. Hyun and Kyu (1997) used the multi agent based scheduling system in shipbuilding yard with pool of dispatching rules. In their approach, dispatching rules are incorporated into agents in order to take logical decisions. Adams et al (1988) used the shifting bottleneck algorithm

to solve the job shop scheduling problem. Gan et al (2002) used the branch and bound approach to solve the scheduling problems in mould manufacturing shops. Heuristic and meta heuristic methods are most popular approaches in current scheduling research. Heuristic methods take the advantage of dispatching rules by combining three or four rules and using them to solve scheduling problems, so chances of bottleneck situations are considerably reduced in heuristic methods. Chu et al. (1998) proposed a heuristic algorithm for job shop scheduling problem, which gradually improves a given solution by reversing the order in which some tasks are performed on machines, which is equivalent to reversing the direction of a critical disjunctive arc. Meta heuristic algorithms such as genetic algorithm, simulated annealing algorithm, menetic algorithm, and tabu search evolve the initial solution to number of generation cycles in order to improve the solutions. Ponnambalam et al. (2000) used the tabu search algorithm to schedule the job shop scheduling problems. They demonstrated the importance of adjacent pair wise mechanism in scheduling problem under job shop environment. Cheng et al. (1995) proposed a genetic algorithm method to minimize the total earliness and tardiness in parallel machine scheduling system. They introduced the job based coding scheme to reduce the solution space in parallel machine system. The detailed review and functions of some of the best meta heuristic algorithms are presented in the subsequent sections.

2.2 PROCESS PLANNING

Process planning is an important task in manufacturing environment. Process planning can be defined as the systematic determination of the detailed methods by which parts can be manufactured efficiently. For example, a detailed mould manufacturing process planning includes the task of process selection and sequencing, machine selection, cutting tool selection, cutting parameters selection, jigs and fixtures selection, etc. In

general, the inputs to process planning are design data, raw material data, facilities data and quality requirement data. Both design and quality requirement data are defined and specified at the mould design stage, based on the specifications and requirements of the product.

Process planning is an important planning tool which is used to bridge most of the planning tasks (Alting and Zhang, 1989). Computer Aided Process Planning (CAPP) is used to integrate the Computer Aided Design (CAD) and Computer Aided Manufacturing (CAM). Over the decades CAPP is main research topic in manufacturing Automation. There are many approaches used in CAPP systems, such as variant approach, generative approach, Artificial Intelligence based approach. Variant approach works by retrieving the plan for a similar part and manually modifying the plan to fit the part at hand to provide the new process plan. Variant approach uses the group technology for classifying and coding parts for the purpose of segregating these parts into family groups. Part model is used to build the process plan in generative process planning system. In generative process planning approach, plans are developed from scratch. Artificial intelligence approaches are used in process planning for last two decades. Mostly Case-based Reasoning (CBR) and Expert system approaches are used in computer aided process planning systems. Case based reasoning works by extracting the analogical reasons from earlier experience and uses them to solve the current problems. In process planning, case based reasoning uses the existing process plans to develop the plan for new part.

In general, most of the process planning systems are constrained by number of assumptions such as unlimited resources and idle shop floor. These assumptions create the problems while executing the process plans in shop floor. Small changes in the shop floor condition makes the process plans to be impossible to follow. As a result, the pre-

generated process plans tend to decrease the overall production performance of the shop. Flexibilities exist in process plans can be used to solve this problem by generating alternative plans, and using them when bottleneck situation arises.

2.3 META HEURISTIC ALGORITHMS

Scheduling problems are formulated to solve and improve certain performance measures, such as makespan, flow time and lateness. Solution space is an important consideration to almost all scheduling problems. Many scheduling problems contain combinatorial solution space. However, number of feasible solution increases with increase in number of jobs and machines in the system. In job shop case, an m machine, n job problem has $(n!)^m$ possible schedules, a phenomenon called “combinatorial explosion”. Combinatorial explosion thus is a critical difficulty in solving scheduling problems. The other difficulty in scheduling is the diversity of conflicting constraints posed by due dates, cost limits, machines, and order characteristics, etc.

Perhaps the most celebrated analytical result in machine shop scheduling for which a theoretical proof exists for its optimality is Johnson’s rule. Johnson’s rule optimally solves the 2-machine flow shop makespan minimization problem and for special situations it may be extended to the 3 machine flow shop. Beyond Johnson’s rule, few general and simple-to-apply results exist. Based on the complexity of the algorithm used to solve optimization problems, all problems are classified into two classes called P and NP. Mostly class P problems are considered easy than class NP problems. Class P problems provide the solution in polynomial time, whereas for class NP problems, it is hard to find polynomial time solution. However, a widely held conjecture of modern mathematics is that there are problems in NP class for which algorithms with polynomial time complexity will never be found. These problems are classified as NP-Hard problems. Unfortunately, most of the practical scheduling problems belong to the NP-Hard class.

From 1960, traditional optimization techniques have been used to solve a few such problems. There are instances in which mathematical programming and intelligent enumeration methods based on branch and bound or dynamic programming have reduced the analyst's computational burden in reaching optimal solutions. Most often, however, the size or the complexity makes it necessary for one to resort to heuristic techniques. Such technique can obtain a solution to a large problem with in the limited computational effort. However, in these methods, optimality of the achieved solution is always been a question. In many such cases the computational requirements are predictable for problems of a given size. However, the primary drawback of heuristic methods is that they do not guarantee providing an optimal solution to all the problems.

Still heuristic job or machine scheduling methods are very successful when no other method works. For instance, the guided local search method by Balas and Vazacopoulos (1998) is claimed to solve job shop problems involving 100 jobs and 20 machines with in 0.5% of optimality in few minutes. Broadly heuristic methods are of two types: (1) Those that limit the space of search by only considering schedules that meet some specified criteria. (2) Those that search in a limited neighborhood of some known feasible schedule to improve the solution at hand. Extensive recent research on the evaluation of the performance of heuristics on a variety of scheduling problems has given some additional promise to their validity. However, it was observed that most the work often considers heuristic's worst case performances, which is not necessarily a true measure of goodness of the utility of a given heuristic (Lenstra et al, 1977).

In problems involving stochastic data, combinatorial techniques may be used. The simpler problems often take advantage of queuing theory results or branch and bound techniques but, by and large, Monte Carlo simulation becomes the technique of frequent choice. Often the complicated environment or simply the highly combinatorial nature of

the problem is enough to render simulation the only viable alternative for an attempt at the solution. More recent advances in this optimization and computational field are Meta heuristic methods. Meta Heuristic methods help to conduct directed intelligent search on very large solution space. It has brought the new possibilities to our ability to search for efficient and economic schedules.

2.3.1 Genetic Algorithm

Over the last decade, genetic algorithms have been extensively used as search and optimization tools in various problem domains, including the sciences, commerce and engineering. Genetic Algorithm (GA) is motivated by the principles of natural genetics and natural selection. Some fundamental ideas of genetics are borrowed and used artificially to construct search algorithms that are robust and require minimal problem information. The working principle of GA is different from most classical optimization techniques. Genetic algorithm starts with the random population, which contains set of chromosomes, representing the solution to the problem. Each chromosome in the population is crossed based on crossover probability in order to generate offspring. Further, the offspring are mutated to adopt the certain features of the parent based on the mutation probability. Finally, selection mechanism is used to create the new population for the subsequent generation.

2.3.2 Simulated Annealing Algorithm

Simulated annealing is an optimization strategy, specially designed to search optimal configuration of the states of the system by guiding the search procedures to escape from the trap of local optimum. The concept of the simulated annealing approach evolves from the physical annealing of solids. The algorithm works by unconditionally accepting the initial solution which results in small energy values than the previous solution. In

subsequent generations, new solutions are either accepted or rejected based on the value of probability function. The ability to occasionally accept degenerate solutions is what separates simulated annealing algorithm from other gradient-descent methods. In gradient descent methods, once a local minimum of the objective function is reached, there can be no further improvements.

The simulated annealing algorithm consists of two iterative loops. Inner loop changes the configuration state of the system, while the out loop maintain the control condition. The temperature of the system is reduced after each generation. The algorithm is iterated until it reaches the final temperature or fixed number of generation cycle. Lot of researchers used this optimization technique to solve their problems. Ma et al (2000) used the simulated annealing approach to search the solution space in job shop environment. Brown and Cagan (1997) used the simulated annealing approach in process planning system. In their approach new form of simulated annealing algorithm named Generative Simulated Annealing is applied to select optimal process plan of prismatic parts.

2.3.3 Tabu Search

Tabu Search (TS) was introduced and improved by the efforts of Glover (1989, 1990). Many computational experiments have shown that tabu search has now become an established approximation technique, which can compete with almost all known techniques, and can beat many classical procedures by its flexibility. TS algorithm is an iterative improvement approach designed to escape from local optimum. Like SA, TS uses the neighborhood mechanism to move from one region of the search to another in order to look for a better solution. When a solution is stuck at a local optimum, SA attempts to escape from it by accepting an inferior solution, which may lead to better solutions later. In contrast, TS allows the search to move to the best solution among the

set of candidate moves as defined by the neighborhood structure. However, subsequent iterations may cause the search to move repeatedly back to the same local optimum. In order to avoid cycling to some extent, moves that would bring back to a recently visited solution should be forbidden for certain number of iterations. This is accomplished by keeping the attributes of the forbidden moves in a list, called a tabu list. The size of the tabu list must be large enough to prevent cycling, but small enough not to forbid too much moves. This systematic use of memory is an essential feature of tabu search.

2.3.4 Memetic Algorithm

Memetic Algorithm is population-based approach which has shown that they are orders of magnitude faster than traditional genetic algorithms for some problem domains. Basically, they combine local search heuristics with crossover operators. For this reason, some researchers have viewed them as hybrid genetic algorithms. However, combinations with constructive heuristics or exact methods may also belong to this class of Meta heuristics. Since they are most suitable for parallel computers and distributed computing systems (including heterogeneous systems) as those composed by networks of workstations, they have also received the dubious denomination of Parallel Genetic Algorithms. Parallel Genetic Algorithm is also described as Genetic Local Search by some of the researchers.

Moscato (1989) introduced this hybrid genetic approach, which combines the recognized strength of population based methods with intensification capability of local search. In Memetic algorithm, all individuals of each population evolve solutions until they become local minima of certain neighborhood. There are two main generic operators used in this algorithm, crossover and mutation. Usually, the crossover is used as main genetic operator and the performance of any genetic system is greatly influenced by this operator.

2.4 INTEGRATION OF PLANNING ACTIVITES

The planning of manufacturing industry includes a number of complex activities. There are two main activities; process planning and scheduling that controls most of the planning activities in manufacturing system. Integrated process planning and scheduling, as the name implies, involves the addition of process planning to the scheduling problem as another dimension or vice versa. There are few attempts made in the past to achieve the integration of scheduling and process planning activities, but the results are not as efficient as expected. There is always a problem in reaching the optimal or near optimal solution in the integrated system. Mostly the problems occurred by the pre-generated process plans. Most of the pre-generated process plan does not provide flexibility in operation sequences, hence chance of improving or integrating the process plan with other planning activities decreases considerably. Looking at the scheduling point of view, the addition of flexible process plan would mean that each job could have chances of obtaining better solutions from the added flexibility. From the perspective of process planning, a glimpse into the loading situation of the shop floor will enable process planners to judge the availability of the machines involved and help prevent bottleneck situations. MADEMA (Manufacturing Decision Making) is one of the first systems, which details the issues in the integration. The assignment of various factory resources to the production tasks are identified as the common aspects of process planning and scheduling functions in MADEMA. The integration problem is modeled as a multiple attribute decision making problem. A decision matrix is formed where the rows represent alternatives while the columns represent the attributes. The choice of one or the other alternative resource is made by the evaluation of its relevant contributions to some established criterion.

ICAPPS (Integrated Computer Aided Process Planning and Scheduling System) is an automated CAPP system based on group technology concept. Along with group technology concept the ICAPPS system is combined with a group scheduling algorithm called key machine loading (KML). A key machine is the one that is loaded more than the others. IPPM (Integrated Process Planning Model) is one of the efficient approaches used to integrate the process planning and scheduling activities which also uses a decision matrix to represent the integration problem. In this approach a fuzzy set operation to select set-ups and machine tools is introduced. All set ups are expressed in their degree of membership to a particular set. Based on the contribution of each set up, one set up will be selected for each form all the possible parts to enter into the solution space in the matrix. Each one of the selected set ups will be assigned to the available machines based on the principle of shortest processing time (Aldakhilallah and Ramesh, 1999).

Like this, there are many of approaches which tried to make the process planning and scheduling systems as a single unit, but the final outcome either became a conceptual system with lot of assumptions or mere interface than integration of these systems. The main problem faced by most of the systems is obtaining the optimum solution in the integrated system (Faruk and Constantin, 1997).

There is much more need for the system or method to find the optimal and applicable solution which can be implemented to the real time system. For example in mould making industries, the need for integration of process planning and scheduling is claimed to be very important for smooth and efficient running of the plant. Nowadays, mould manufacturers often undertake the design, manufacture, testing and even the production of their client’s product. The mould manufacturer from whom the testing data are taken does not operate a pure flexible manufacturing system. They instead have a collection of

machines, which are grouped according to machine type and workers do the movement of jobs manually. The process plans gives a sequence of machining department that a part has to go through before it is completed. The jobs are then passed to the production shop floor, which follows the process plan to manufacture the part. On the shop floor, each machining department operates independently by scheduling the parts queuing on the machines and passing the machined parts to the next respective departments. Developing the suitable integrated process planning and scheduling approach for this environment will solve the most of the planning problems in mould manufacturing shop. Before developing the integrated system, suitable algorithm and searching rules have to be designed for better integration. To achieve this, process planning department has to list out constraints between operations instead of simply fixing the required operations. The new approach should use the flexibilities in process plans to generate the better schedule, while the constraints have to be followed to sustain the feasibility of the solution.

CHAPTER 3

HEURISTIC ALGORITHMS

3.1 INTRODUCTION TO HEURISTIC ALGORITHMS

A heuristic approach to a problem is an empirical search or optimization method that often works at solving the problem, but does not have any of the rigorous proof that people like physicists and mathematicians expect. As a rule, scheduling problem formulations engage some economic performance criteria. Many scheduling formulations seek combinatorial solutions. However, the number of feasible schedules in combinatorial optimization grows rapidly with the number of jobs, machines, processing steps, etc., making almost all problems of practical significance difficult to solve optimally.

In general, heuristic methods are of two types: (1) Those that limit the space of search by only considering schedules that meet some specified criteria. (2) Those that search in a limited neighborhood of some known feasible schedule to improve the solution at hand. Extensive recent research on the evaluation of the performance of heuristics on various scheduling problems has given some additional promise to scheduling research. However, it was observed that most of the studies often consider only a heuristic's worst case performances, which is not necessarily a true measure of goodness of the utility of a given heuristic. Few heuristic algorithms are available to solve the solution time problems in single machine and flow shop scheduling problems. However, polynomial time algorithm to sequence and schedule the jobs in parallel machine shop is not concentrated much. In order to solve these problems in parallel machine shops, two heuristic algorithms are developed with the aim of solving the planning problems in polynomial time.

3.2. IDENTICAL PARALLEL MACHINE SHOP

In the manufacturing world, parallel machine shops are considered very important for smooth flow of production process. Most of the manufacturing industries are working with shops of parallel machines from small scale to large scale sectors, so it became an important concept of research since its introduction. In scheduling problems, it is often possible to take advantage of parallelism in resource structure.

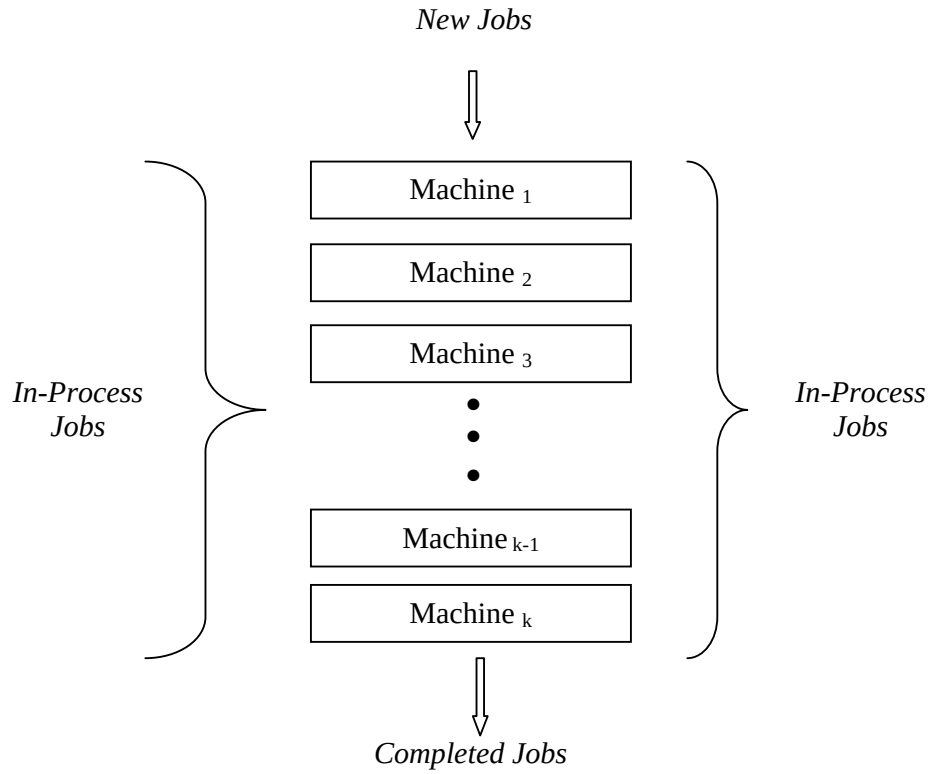

Figure 3.1 Work flow at a typical single stage parallel machine shop

In the typical parallel machine shops, there are *n* jobs available in front of the machines with processing time t_i and due date d_i. They have to be processed on continuously available identical parallel machines with the objective of the scheduler. Pre-emption is not allowed because disturbance in single operation will greatly affect the quality of the product, which will also increase the lead-time. This kind of identical parallel machine shops can be frequently found in mould manufacturing industries, where the plant is

divided into a serious of parallel machine manufacturing shops in which jobs are passed between them for processing.

The following assumptions are considered while modeling the heuristic techniques.

1) The jobs are independent and only need to be processed once.
2) Each machine can process only one job at a time.
3) All processing time includes the loading and unloading times.
4) Preemption is not allowed in the jobs

Consider the following nomenclatures for parallel machine environment

t_i = Processing time of job i

d_i = Due date for job i

S_i = Slack time available in job i ; $S_i = d_i - t_i$; for i = 1, 2…. *n*

$\{L_s\}$ = Local set of jobs (conditional set of jobs)

$\{G_s\}$ = Global set of jobs (confirmed set of jobs)

$\{k_j\}$ = Set representing instances of processing times

TL_i = lateness of job i; $TL_i = C_i - d_i$

TR_i = Tardiness of job i

C_i = Completion time of job i

EL_i = Earliness of job i

$TR_i = \max\{0, L_i\}$; $EL_i = \max\{0, d_i - C_i\}$

With these descriptions, heuristic algorithms are developed to maximize the number of in-time or early jobs in parallel machine shop. In scheduling systems, maximizing the number of in-time and early jobs indirectly minimizes the number of tardy jobs. Thus, this system can be expressed as the minimization of $P_m \parallel \sum_{j=1}^{n} X_j^T$. This classical formulation of minimizing the number of tardy jobs in parallel machine scheduling is known as NP-hard problems.

3.3. HEURISTICS FOR PARALLEL MACHINE SHOP

In the aim of developing polynomial time algorithm for planning problems in parallel machine scheduling shop, two heuristic techniques are proposed in this chapter. The complexity of planning tasks increases with increase in number of jobs and machines. Even small increase in number of jobs or machines increases the scheduling permutations heavily (Koulamas, 1994). The first heuristic (HC_I) is designed to give polynomial time solution for maximizing the number of in-time jobs, while the second one (HC_{II}) is designed to provide better solution for sequencing and scheduling problems in parallel machine shops.

In the first heuristic (HC_I), jobs are sorted based on S_i (Slack) and scheduled to respective machines with the sequence, so scheduling and sequencing are considered simultaneously. However, in the second heuristic (HC_{II}), sequencing and scheduling are considered separately by two phases. The Phase I deals mainly the scheduling of jobs followed by Phase II with sequencing of jobs to each machines.

Heuristic I (HC_I):

STEP 1:

1) Arrange jobs in increasing order of S_i for i = 1,2…*n*; $S_i = d_i - t_i$;

IF (tie occurs)

Assign the smallest processing time job first

2) Number the jobs based on current order

3) Let $\{ k_j \} = \{ t_j \}$ for j = 1,2…. *m*; *m* = no of machines

$\{L_s\} = \{J_{1, 2 \ldots m}\}$; a = *m*; $\{G_s\} = \{\{1\}, \{2\} \ldots \{m\}\}$

STEP 2:

a = a + 1; a = counter

IF (min $\{k_j\} \leq S_a$)

Then

$\{L_s\} = \{L_s\} \cup \{J_a\}$

$\{G_s\} = \{G_s\} \cup \{j_i\}$

$k_j = k_j + t_a$

IF (a=*n*)

Go to **STEP 3**

Else

Go to **STEP 2**

End IF

STEP 3:

All Jobs in $\{G_s\}$ are scheduled

Unscheduled jobs will go to **STEP 4**

// This set contains local scheduled jobs, which have tight due date (in-time jobs) or early jobs.

STEP 4:

1) $\{L_s\} = \Phi$

Arrange the unscheduled jobs on increasing order of S_i ; i=1, 2,3…*n*

$\{k_j\} = \{t_j\}$ for j = 1, 2…. *m*

$\{L_s\} = \{J_{1, 2.... m}\}$

a = *m*; *n* = number of jobs in current list

2) IF (*n* >a)

Go to **STEP 2**

Else

$\{G_s\} = \{G_s\} \cup \{L_s\}$

Repeat this procedure until *n*=0

STEP 5:

Set $\{G_s\}$ contains m subsets, assign to particular machines in order.

In this heuristic, step 2 is repeated for n-m times to form an initial local schedule. The scheduled jobs are also sequenced to particular machines simultaneously. Experimental trials shows that the performances of the heuristics are considerably better in large problem space (m>3 and n>15) than smaller ones.

3.3.1 Numerical Illustration for Heuristic algorithm I

In this example, the environment consists of 4 CNC machines and 15 jobs are considered. Each job is ready at the beginning of the scheduling horizon and has a distinct processing time and a distinct due date.

Table 3.1 Details of jobs for HC_I

Jobs	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
ti	5	13	8	3	3	8	10	9	5	10	2	5	11	20	4
d_i	20	26	12	15	20	26	22	20	13	15	5	19	15	39	23
S_i	15	13	4	12	17	18	12	11	8	5	3	14	4	19	19

S_i is the starting time for job i that assures the completion of job in-time ; $S_i = d_i - t_i$

STEP 1:

Arrange jobs in ascending order of S_i

Table 3.1(a) Details of jobs with ascending order of slack time

Jobs	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
ti	2	8	11	10	5	9	3	10	13	5	5	3	8	4	20
d_i	5	12	15	15	13	20	15	22	26	19	20	20	26	23	39
S_i	3	4	4	5	8	11	12	12	13	14	15	17	18	19	19

$\{k_j\} = \{2, 8, 11, 10\}$; $m = 4$;

$\{L_s\} = \{1, 2, 3, 4\}$; a=4; $\{G_s\} = \{\{1\}, \{2\}, \{3\}, \{4\}\}$

STEP 2:

a = 5

Min $\{k_j\} = 2$

$S_a = S_5 = 8$

2<8, So

$\{L_s\} = \{1, 2, 3, \text{and } 4\} \cup \{5\}$;

$\{G_s\} = \{\{1\}, \{2\}, \{3\}, \{4\} \cup \{5_1\}\}$

$\{G_s\} = \{\{1, 5\}, \{2\}, \{3\}, \{4\}\}$

$\{k_j\}= \{7, 8, 11, \text{and } 10\}$;

$5 \neq 15$ Go to **STEP 2**

Table 3.1(b). Procedure to perform step 2 of HC_I

a	(K_j, S_a)	(a, n)	$\{K_j\}$	{Gs}
6	7<11	$6 \neq 15$	{16,8,11,10}	{{1,5,6},{2},{3},{4}}
7	8<12	$7 \neq 15$	{16,11,11,10}	{{1,5,6},{2,7},{3},{4}}
8	10<12	$8 \neq 15$	{16,11,11,20}	{{1,5,6},{2,7},{3},{4,8}}
9	11<13	$9 \neq 15$	{16,11,20,13}	{{1,5,6},{2,7},{3,9},{4,8}}
10	11<14	$10 \neq 15$	{16,16,20,13}	{{1,5,6},{2,7,10},{3,9}, {4,8}}
11	13<15	$11 \neq 15$	{16,21,20,18}	{{1,5,6},{2,7,10},{3,9},{4,8,11}}
12	16<17	$12 \neq 15$	{19,21,20,18}	{{1,5,6,12},{2,7,10},{3,9},{4,8,11}}
13	18=18	$13 \neq 15$	{19,21,20,26}	{{1,5,6,12},{2,7,10},{3,9},{4,8,11,13}}
14	19=19	$14 \neq 15$	{23,21,20,26}	{{1,5,6,12,14},{2,7,10},{3,9},{4,8,11,13}}
15	20>19	15=15	{23,21,20,26}	{{1,5,6,12,14},{2,7,10},{3,9}, {4,8,11,13}}

STEP 3:

$\{L_s\}=\{1,2,3,4,5,6,7,8,9,10,11,12,13,14\}$

$\{G_s\}=\{\{1,5,6,12,14\}, \{2,7,10\}, \{3,9\}, \{4,8,11,13\}\}$

Unscheduled job = 1

STEP 4:

$\{L_s\}= \Phi$

One job unscheduled

Jobs	t_i	d_i	s_i
15	20	39	19

$\{L_s\}= \{15\}$;

$\{20\}= \{k_j\}$: a = 4; *n* =1

a > *n*

$\{G_s\} = \{\{G_s\} \cup \{15\}_1\}$

STEP 5:

m=4;

$\{G_s\}= \{\{1,5,6,12,14,15\}, \{2,7,10\}, \{3,9\}, \{4,8,11,13\}\}$

Sequenced Jobs for Machine 1: 1, 5, 6,12,14,15

Sequenced Jobs for Machine 2: 2,7,10

Sequenced Jobs for Machine 3: 3, 9

Sequenced jobs for Machine 4: 4, 8, 11, 13

The above mentioned heuristic performs the sequencing and scheduling tasks of parallel machine shops in polynomial time. In this heuristic sequencing and scheduling tasks are solved concurrently, so in order to study the effect of separate execution of these two tasks, a new heuristic algorithm is developed. The developed heuristic algorithm (HC_{II}) performs the planning tasks separately.

Heuristic II (HC_{II}):

This heuristic algorithm contains two phases. The first phase does the scheduling part while the second phase performs the sequencing part. In this heuristic, the second step of Phase I is repeated for n-m times to obtain local schedule in polynomial time. In this heuristic, increase in number of in-time jobs is given higher importance than early jobs.

PHASE I

STEP 1:

1) Arrange jobs in increasing order of S_i for i = 1,2…*n*; $S_i = d_i - t_i$;

   IF (tie occurs)

   Assign the smallest processing time job first

2) Number the jobs based on current order

3) Let $\{ k_j \} = \{ t_j \}$ for j = 1,2…. *m*; *m* = no of machines

   $\{L_s\} = \{J_{1, 2 \ldots m}\}$; a = *m*; $\{G_s\} = \Phi$

   $\lceil c \rceil = n/m$

STEP 2:

a = a + 1;

IF (min $\{ k_j \} \leq S_a$) Then

$\{L_s\} = \{L_s\} \cup \{J_a\}$

$k_i = k_j + t_a$

IF (a = *n*) Go to STEP 3

Else

Go to STEP 2

End IF

STEP 3:

All Jobs in $\{L_s\}$ are scheduled

Unscheduled jobs will go to Phase II

// This set contains local scheduled jobs, which have tight due date (in-time jobs) or early jobs.

PHASE II

STEP 1:

1) $\{G_s\} = \{G_s\} \cup \{L_s\}$

   $\{L_s\} = \Phi$

2) Arrange the unscheduled jobs in increasing order of S_j

   $\{k_j\} = \{t_j\}$ for j = 1, 2…. *m*

   $\{L_s\} = \{J_{1, 2.... m}\}$

   a = *m*

3) IF (*n* >a)

   Go to STEP 2 of Phase I

   Repeat this procedure until *n*=0

STEP 2:

1) Split $\{G_s\}$ into *C* Subsets

2) 2a) Sequence $\{SS\}_w$ in ascending order of processing time for w = 1, 3, 5…. *C*

   2b) Sequence $\{SS\}_e$ in descending order of processing time for e = 2, 4, 6… *C*

3) Rearrange the subsets into single set by non-decreasing order of *C*

   3a) Schedule jobs to first m machines

   3b) IF (*n* > *m*)

Start schedule from first machine followed by previously scheduled jobs

3.3.2 Numerical Illustration for Heuristic Algorithm II

For this illustration, 2 CNC machines and 10 jobs parallel machine shop environment is considered with the objective of maximizing the total number of in-time and early jobs.

Table 3.2 Details of jobs for HC_{II}

Jobs	1	2	3	4	5	6	7	8	9	10
t_i	11	18	6	12	21	10	14	28	13	19
d_i	14	40	46	48	46	22	16	38	30	23
S_i	3	22	40	36	25	12	2	10	17	4

S_i is the starting time for job i that assures the completion of job in-time ($S_i = d_i - t_i$)

The detailed numerical illustration of the heuristic algorithm II is presented in the Appendix A. The final schedule and sequence of the jobs obtained using the heuristic algorithm II are presented below.

Final schedule {{1, 2}, {6, 5}, {7, 9}, {3, 10}, {4, 8}}

Sequenced Jobs for Machine 1: {1, 6, 7, 3, 4}

Sequenced Jobs for Machine 2: {2, 5, 9, 10, 8}

3.4 COMPARISON OF HEURISTICS

The proposed heuristics can find the polynomial time solution for sequencing and scheduling tasks in identical parallel machine shops. The parallel machine shop environment is considered with *n* jobs, *m* machines having distinct process times, due dates, and starting times. The heuristics are modeled to obtain the maximum number of in-time and early jobs . The first one (HC_I) solves the planning tasks of parallel machine shops in polynomial time with the importance of increasing the number of in-time and

early jobs. The second heuristic (HC_{II}) is designed to schedule the jobs in parallel machine shop along with sequencing of jobs in each machine. Even though the HC_{II} algorithm is performing the two tasks separately, the performance of the algorithms seems to be similar. When comparing HC_I with Lann and Mosheiov's (2003) MAXONTIME_DD algorithm for minimization of $\sum_{i=1}^{n}(X_i^e + X_i^t)$,where X_i^e and X_i^t represents number of early jobs and tardy jobs, the newly proposed heuristic gives considerably higher number of early and in-time jobs. The proposed heuristics (HC_I and HC_{II}) also considers the sequencing tasks along with scheduling. The jobs which have small slacks are given importance in HC_I, which is considered exactly opposite in MAXONTIME_DD algorithm. In MAXONTIME_DD algorithm, bigger jobs are given higher importance than smaller jobs. From the illustration of algorithm HC_I , it seems that giving higher importance to smaller jobs will increase the throughput of the overall system. With the objective function of increasing the number of early and in time jobs, it will automatically reduce the number of tardy jobs in the system and vice versa. Even though the developed heuristics producing better results for small scale parallel machine shop, there are lot of constraints that have to be dealt with while implementing in real time environment, such as number of operations in jobs, precedence constraints between jobs, and precedence constraints between the operations in job. Thus, these heuristics can not be applied directly to large scale (or) real time industrial system. However, these heuristic techniques can be used to reduce the solution space of the parallel machine system. Considering all these factors and simulation results, a search is directed to find the suitable Meta heuristic technique for scheduling parallel machine shop system.

CHAPTER 4

OPTIMIZATION TECHNIQUES

4.1 THE OPTIMIZATION PROBLEM

Optimization is performed to optimize the new system or improve the existing system. Most cases, optimization is performed for minimizing or maximizing certain objective function with respect to certain constraints. There are a number of optimization techniques available today. Optimization techniques are selected based on the problem environment and solution space of the problem. Conventional optimization techniques have the problem of finding optimal or near optimal solution for non-linear problems. To solve these problems with non-linear tasks, heuristic algorithms are developed based on single rule or combination of rules. However, the problem size became a big problem for heuristic techniques to be implemented in large scale systems. These drawbacks in heuristic techniques are solved by Meta heuristic techniques. This chapter provides the detailed study about optimization techniques used in machine scheduling problems. Along with this, four Meta heuristic algorithms are evaluated by scheduling the single stage parallel machine shop environment. A DOE (Design of Experiment) technique is developed to compare the performances of the Meta heuristic techniques.

4.2 PERFORMANCE MEASURES

Schedules are generally evaluated by aggregate quantities that involve information about all jobs, resulting in one-dimensional performance measures. Measures of schedule performance are usually functions of the set of completion times in a schedule. For example, suppose that n jobs are to be scheduled in system. Aggregate performance measures might be makespan, flow time, tardiness, earliness and number of late jobs.

Each of these measures is a function of the set of job completion times, so that their general form is always

$$\mathbf{Z} = f(C_1, C_2, C_3, \ldots\ldots, C_n) \quad (4.1)$$

Furthermore, these quantities belong to an important class of performance measures that are called ***regular*** measures of performance. A performance measure Z is regular if

a) The scheduling objective is to minimize Z, and
b) Z can increase only if at least one of the completion times in the schedule increases.

If the performance measure does not follow the above mentioned criteria, it is called non-regular measures. Performance measures such as minimization of total lateness and maximization of machine utilization are some of the good non-regular performance measures to be considered in real time scheduling. Unlike regular measures, non-regular measures are little difficult to investigate and it varies based on the system. The following quantities are considered while dealing with scheduling systems,

Processing time(t_i) : The amount of processing required by job i.

Ready time (r_i) : The point in time at which job i is available for processing.

Due Date (d_i) : The point in time at which the processing of job i is due to be completed.

Completion time (C_i) : The time at which the processing of job i is finished.

4.2.1 Makespan

The completion time of all the jobs in the system is formally defined as the makespan of the system. But the calculation of makespan varies from system to system. In the case of single machine scheduling problem, it is the completion time of all jobs in the system. Makespan in parallel machine system is defined as the maximum of completion times of last completed job in each machine of the system.

In the case of single machine system, makespan is calculated as follows.

$$M^*_{\text{Single machine}} = \sum_{i=1}^{n} C_i \tag{4.2}$$

C_i = Completion time of i^{th} job in single machine system

In single stage parallel machine scheduling , Makespan is calculated as follows.

$$M^*_{\text{Parallel machine}} = \max_{k=1}^{m} \left\{ \sum_{i=1}^{n} C_{ik} \right\} \tag{4.3}$$

4.2.2 Flow time

Flow time is another important measure used to quantify the performance of the scheduling system. It is generally defined as the amount of time that job i spends in the system. It is simply denoted as $F_i = C_i - r_i$. Generally the total flow time of the single machine system is calculated by,

$$F^* = \sum_{i=1}^{n} C_i - r_i \tag{4.4}$$

In a machine based parallel machine system, the flow time of the system is calculated by:

$$F^* = \max_{k=1}^{m} \left\{ \sum_{i=1}^{n} C_{ik} - r_{ik} \right\} \tag{4.5}$$

4.2.3 Lateness

Lateness is the one of the very important measure which will directly affect the profit and good will of the organization. Thus, much care is given to minimize this important measure in manufacturing industries. Lateness(L_i) of job i is defined as the amount of time by which the completion time of job i exceeds its due date. Lateness measures the conformity of the schedule to a given due date. Lateness measures consist of two dimension namely negative lateness and positive lateness. Negative lateness represents better service than requested, while positive lateness represents poorer service than requested. In many situations, penalties and added costs will be associated with positive

lateness, but no benefits will be associated with negative lateness (Balakrishnan et al., 1999). In some situations, negative lateness also indicates the poor performance of the system, such as idle time of the system, and under utilizations of the recourses. However, the importance to reduce the negative lateness is comparatively smaller than positive lateness, so different penalties and costs are considered while dealing with different lateness measures.

Some of the lateness related measures are explained below:

Tardiness (T_i): The lateness of job i if it fails to meet its due date, or zero.

Otherwise: $T_i = \max \{0, L_i \}$

Earliness (E_i): The negative lateness of job i or zero.

Otherwise: $E_i = \max \{0, | L_i |\}$

In addition to commonly used lateness measures, priority added measures are also considered in this research. Based on the system environment priority values are assigned to the jobs. While studying the single stage parallel machine systems priority values are assigned from scale 1-6, whereas in large scale system studies priority value of 1-10 is assigned to job. However, priority values are fixed based on the problem environment. In lateness measures, penalty value (β) is assigned to early jobs whereas penalty value (α) assigned to tardy jobs. The penalty value decides the impact of particular measure on the system.

$$WEL_i = (d_i - C_i) \times P_i \times \beta \quad (4.6)$$

$$WTR_i = (C_i - d_i) \times P_i \times \alpha \quad (4.7)$$

4.3 ONE PASS OPTIMIZATION TECHNIQUES

The details about one pass optimization techniques and simple heuristic techniques which can be applied to scheduling problems are discussed in this section. One pass optimization techniques generates the solution to the given optimization problem in

single pass of the computer program. One-pass optimization techniques work based on the utilization of machines in the system. Logic incorporated in one pass optimization technique decides the schedule for every incoming job. The problem arises whenever bottleneck situation occurs in the queue.

4.3.1 Dispatching rules

Dispatching rules plays an important role in sequencing and scheduling problems. Lots of researches have been done under this topic and some of the rules are still proved better choice for certain planning condition (Blackstone et al., 1982). SPT (Shortest Processing Time) rule is one of the best dispatching rules to effectively schedule the jobs by minimum mean flow time in single machine system. Dispatching rules are mostly used in parallel machine computer clusters, however modern automation concepts uses this technique in manufacturing industries. Baker (1974) developed some of the best dispatching rules for machine scheduling problems. Inherent problems in dispatching rules (such as lot of assumptions while designing the rules, single logic to decide the jobs from queue and etc) decrease the chance of applying the dispatching rules to large scale systems.

4.3.2 Simple Heuristic Techniques

Some of the heuristic methods are still showing the promising results in scheduling problems. For example, Johnson method is the best method to solve two-machine flow shop scheduling problem. The two-machine flow shop problem, with the objective of minimizing makespan can be optimally solved by Johnson's heuristic method than other optimization and mathematical techniques.

In job shop scheduling, the Giffler and Thomson heuristic (Giffler and Thompson, 1960) is proved as one of the best techniques to use. But there are very few heuristic which can

deal with parallel machine scheduling systems. Even the heuristics which are generated for solving parallel machine scheduling machine, as described in the previous chapter are only suitable for small scale systems. Heuristic methods are applicable only in hypothetical systems and deterministic systems. Based on the study and review, it seems that the single heuristic algorithms are not efficient to solve and improve the planning tasks of entire manufacturing system.

4.4 META HEURISTIC TECHNIQUES

A typical scheduling problem comprises several concurrent and often conflicting goals and a multitude of resources available to satisfy those goals. It is noted that a combination of several goals and resources may result in an exponentially growing possible solution space. In such cases, it becomes difficult or even impossible to find exact solutions in reasonable time. Many scheduling problems seek combinatorial solutions. However, the number of feasible schedules in combinatorial optimization grows with the number of jobs, number of machines, number of processing steps, etc., making almost all problems of practical significance difficult to solve optimally.

To solve these scheduling problems in manufacturing shops, Meta heuristic methods are used. Generally Meta heuristic methods are methods that help to conduct directed "intelligent" search in the potentially very large solution space. These methods have brought new possibility to search for efficient and economic schedules. There are number of Meta heuristic methods available today. The four of those best methods are chosen to study thoroughly, in order to implement them in large scale system. The Meta heuristic methods which are studied in this chapter are Genetic algorithm, Simulated annealing, Memetic algorithm and Tabu search.

4.4.1 Genetic Algorithm

Genetic algorithm is one of the problem solving systems based on the principles of evolution and hereditary. It starts with initial solutions and uses a process similar to biological evolution to improve upon them that encourages the survival of the fittest. The best overall solution becomes the candidate solution to the problem. It is best described as GA is based on evolution theory that the genes of superior individuals in a population gets to progress while inferior genes are slowly phased out over many generations. GA processes a number of solutions in a population all at the same time and advances the quality of these solutions with each generation. Thus, it effectively gives a selection of solutions as the final output and is feasible enough to allow customization on configuration in many ways.

To begin, a starting population of solution is randomly created and the objective function, usually the performance measure, is used to calculate the fitness of each solution. As a result, the objective function is also called the fitness function in GA. Same number of solutions will be generated for the new population by crossover and mutating the existing pool of solutions. The chance for each solution to be copied for crossover and mutation is based on its fitness values. Solutions from the existing population that are selected for crossover and mutation are called parent solution while those that have undergone the change are called offspring solutions. The reproductive selection procedure picks the solutions from the pool for the next generation giving priority to the fitter solutions. The next round of cycle will begin until a certain termination condition is achieved. Usually, the algorithm can be set to terminate after a number of cycles are achieved or when there are no improvements after a number of cycles. When the program is successfully terminated, the final population of solutions will converge towards the optimum. The pseudo code of implemented genetic algorithm

STEP 1: Set evolution environment, such as *pop_size*, p_c, p_m and *max_gen*.

STEP 2: Generate an initial population containing

Pop_size chromosomes by Heuristics

STEP 3: Make *pop_size**p_c offspring using proposed procedure

STEP 4: Make *pop_size**p_m offspring using proposed procedure

STEP 5: Calculate fitness for expanded populations make a *roulette wheel*.

Spin it *pop_size* times to get next generation.

STEP 6: Select the best population from current and extended populations.

STEP 7: If generation equals to *max_gen*, stop the evolutionary process;

Otherwise return back to **STEP 2**

Figure 4.1 Pseudo-code for implemented Genetic Algorithm

4.4.2 Simulated Annealing Algorithm

The simulated annealing algorithm was derived from statistical mechanics. Kirkpatrick et al. (1983) proposed an algorithm, which is based on the analogy between the annealing of solids. Its basic thought is to apply randomness of the algorithm and to increase freedom of optimization of the algorithm. It would also accept bad solution with a certain probability, thus escaping from local optimum and tending to global optimum. The general structure of simulated annealing algorithm is given below:

STEP 1

Randomly generate a feasible schedule called the Current schedule

STEP 2

Start from the initial temperature $\mathbf{T=T_0}$,

While not reaching the final temperature $\mathbf{T_{lowest}}$

{

a. Make a random change to the current schedule, let temp-schedule be the schedule after change.

b. Check to make sure that current schedule is valid. Otherwise, go back to (a)

c. Calculate the objective value of current schedule (F_1) and temp-schedule (F_2).

If (F_2<F_1)

Let temp-schedule be current schedule

Else

Randomly generate X (0<X<1)

If(X< $e^{(Delta)/T}$); Delta=F_1-F_2

Let temp-schedule be current schedule

Else

Let current-schedule remain unchanged

End if

End if

d. Repeat (a) to (c) until a criterion is satisfied.

f. Reduce the temperature to a new **T**

}

Figure 4.2 Pseudo-code for implemented Simulated annealing algorithm

4.4.3 Memetic Algorithm

Moscato and Norman have introduced the term memetic algorithm to describe the genetic algorithm in which local search plays a significant part (Moscato, 1989). In memetic algorithm, a local optimizer is applied to each child before it is inserted into the population in order to push that to climb its local optima. With local climb heuristic, this hybrid genetic algorithm is used to perform global exploration among population while

heuristic methods are used to perform local exploitation around chromosomes. Due to the complementary properties of genetic algorithms and conventional heuristics, the hybrid approach often outperforms either method when operating alone. The general structure of memetic algorithm is described below,

```
Begin
    t← 0;
    Initialize P (t);
    Evaluate P (t);
    While (not termination condition)
     Do
        Begin
            Recombine P (t) to yield C (t);
            Locally climb C (t);
            Evaluate C (t);
            Select P (t+1) from P (t) and C (t);
            t← t+1;
        end
End
```

Figure 4.3 Pseudo-code for implemented Memetic algorithm

P (t) and C (t) are parents and offsprings in current generation t. The solution space is constructed on job based and machine based coding schemes.

4.4.4 Tabu Search

Tabu search (TS) is a meta-heuristic that guides a local search to explore the solution space beyond local optimality (Glover, 1989, 1990). Tabu search has now become an established approximation technique, which can compete with almost all optimization

techniques, and is one of the classical procedures for its flexibility. It consists of several elements called the move, neighborhood, initial solution, search strategy, memory, aspiration function and stopping rules. An aspiration function is introduced in tabu search to determine when tabu restriction can be overridden, thus removing a tabu classification otherwise applied to move. The tabu tenure period is the number of subsequent moves during which the last pair of solutions to be forbidden. The rule considered for this study uses static rule in which tabu tenure t = *n*, (where n is number of jobs) is used.

The general structure of tabu search is given below,

STEP1

X=1;

Select an initial solution S_I using some heuristic and set $S_{best} = S_I$.

STEP 2

Select $S_c \in N(S_k)$

If the move $S_k \rightarrow S_c$ is prohibited by a move on the tabu matrix than go to **STEP 2**

If the move $S_k \rightarrow S_c$ is not prohibited by a move on the tabu matrix then set $S_{k+1} = S_c$

Enter the move at the tabu matrix. Reduce the tabu elements by one

If $F(S_c) < F(S_{best})$

Then

$S_{best} = S_c$ then go to **STEP 3**

STEP 3

X=X+1;

If stopping condition is true

Then STOP

Else go to **STEP 2**

Figure 4.4 Pseudo-code for generalized Tabu (short-term) algorithm

4.5 COMPARISON OF META HEURISTIC METHODS

4.5.1 Problem statement and formulations

In this study, the identical parallel machine scheduling problem for minimizing the total lateness with and without the consideration of priority is represented as follows: There are *n* independent jobs and *m* identical machines available in the system. Each job has its fixed processing time and due date. The job can be completed by either of the available machines in the system. Priority weighted lateness measures are considered in the study. The earliness and tardiness are considered with different penalty factors.

Lateness measures the conformity of the schedule to a given due date, and it is important to note that the lateness quantity takes on negative values whenever a job is completed early and positive values whenever a job is completed tardy. The objective functions of this problem can be represented as follows:

Without priority:

$$\min_{\sigma \in \prod} \boldsymbol{f}(\boldsymbol{\sigma}) = \sum_{j=1}^{n} (\alpha E_j + \beta T_j) \tag{4.8}$$

With priority:

$$\min_{\sigma \in \prod} \boldsymbol{f}(\boldsymbol{\sigma}) = \sum_{j=1}^{n} w_j (\alpha E_j + \beta T_j) \tag{4.9}$$

4.5.2 Representation of solution seed

Solution seed for this study is represented in two different schemes, machine based coding scheme and job based coding scheme. In *machine code scheme*, each seed is represented as A_1, $A_2 \ldots A_j \ldots A_n$. Where $A_j \in [1, m]$ and positive integer number. In *job code scheme*, each seed is represented as B_1, B_2 $\ldots B_j \ldots B_n$, where $B_j \in [1, n]$ and non-repeatable positive integer number. For example, one of the solution seed for 3 machines 6 jobs parallel machine problem in *machine code system* is 1 3 2 2 3 1, which represents

that the first job is assigned to first machine and the second job to third machine and so on, whereas in *job code system,* the solution seed is represented as 3 4 5 6 2 1. Unlike the machine code system, job code system gives the sequence of jobs. Machine code system will give the sequence and schedule for jobs, where as the job code system will just give the schedule of jobs. After scheduling, sequencing has to be performed by some heuristic rule or simple logics.

4.5.3 Parameters selection

The processing time, due date and priority of each job is taken from a local mould manufacturing shop. Based on the source data, further data sets are generated. Initially the tardy penalty *(α)* and early penalty *(β)* is assigned to 8 and 2 respectively . The numbers of jobs that are early or late are recorded in each run to see the effects of α and β on the quality and nature of solution. In most of the experiments, α values greater than 7 do not have considerable change to the solutions. Smaller problems are found to be more insensitive to β. To give higher importance to tardiness than earliness, tardy jobs are assigned with penalty value of 6 while early jobs are assigned with penalty value of 2 (Biskup and Cheng, 1999).

4.5.3.1 Crossover

This operation is considered as the one that makes the evolutionary algorithms different from other algorithms, such as dynamic programming. It is used to create one or two new individuals from two existing individuals picked from the current population by the selection operation. There are several ways to perform crossover operation in genetic algorithm. Position based crossover operator is used for both the memetic and genetic algorithm for job code system, whereas machine code systems are equipped with two point crossover operator. Crossover probability of 0.80 is chosen for all the studies.

Working procedures of position based and two point crossover operators are given in Figure 4.5,

	Position Based Crossover	Two Point Crossover
	↓ ↓	↓ ↓
Parent 1:	1 2 3 4 5 6 7 8 9 10	1 2 3 4 1 2 3 1 4 1
Parent 2:	10 9 8 7 6 5 4 3 2 1	2 3 4 1 1 3 2 1 2 4
Offspring 1 :	10 9 3 8 6 5 7 4 2 1	2 3 4 4 1 2 3 1 2 4
Offspring 2:	1 2 8 3 5 6 4 7 9 10	1 2 3 1 1 3 2 1 4 1

Figure 4.5 Working processes of position based and two point crossovers

4.5.3.2 Mutation

In general mutation helps to sustain the better genes in offsprings. The mutation operator forces the algorithm to search new areas. It also helps to avoid premature convergence of solution. Swap mutation is applied to both the job and machine coded systems of GA and MA. Swap and inverse mutation operators are randomly chosen in SA. These operators are given importance based on the searching scheme of solution space. The working principle of swap and inverse mutation operators are described in Figure 4.6 ,

Swap mutation

	↓ ↓
Parent :	1 2 3 4 5 6 7 8 9 10
Offspring :	1 2 7 4 5 6 3 8 9 10

Inverse mutation

	↓ ↓
Parent :	1 2 3 4 5 6 7 8 9 10
Offspring:	1 2 7 6 5 4 3 8 9 10

Figure 4.6 Working processes of swap and inversion mutations

4.5.3.3 Selection scheme

The aim of the selection procedure is to reproduce more copies of individuals whose fitness values are higher. The selection procedure has a significant influence on driving the search towards a promising area and finding good solutions in a short time. However, the diversity of the population must be maintained to avoid premature convergence and to reach the optimal solution. Roulette wheel selection scheme is used for the selection mechanism. Roulette wheel selection scheme reproduces the subsequent generations based on current enlarged population, in which a fitter chromosome has a higher chance to be reproduced into next generation.

4.5.3.4 Creation of Initial Solutions

At the start of optimization, all meta-heuristic algorithms require an individual or group of initial solutions. There are two ways to create this initial population. One method is randomly generating initial solution using random number generator. This method is preferred for problems about which no priori knowledge exists for assessing the performance of an algorithm. The second method employs a priori knowledge about the given optimization problem. The former method is used for initial population selection in most of the studies here. However, some modifications are made in the randomly generated initial population in order to sustain the feasibility of the solution.

4.5.3.5 Size of subneighbourhood

The population in GA and MA is considered as a static size of 10, whereas in TS, adjacent pair wise interchange is used to create the neighborhood structure. The number of solutions in the neighborhood is fixed to $n-1$, where n is the problem size.

4.5.3.6 Intermediate and Long term memory strategies

These strategies are mainly used in tabu search algorithm. The intermediate function provides an element of intensification. It operates by recording good features of a selected number of moves generated during the execution of the algorithm. This can be considered as a learning strategy, which seeks new solutions that exhibit similar features to those previously recorded. This is achieved by restricting moves that do not possess favorable features.

4.5.3.7 Termination condition

The termination condition used in this study is a fixed number of generation cycles, and it is preset as 1000 cycles. Even though the simulated annealing algorithm is controlled by cooling factor, the generation cycles of the simulated annealing algorithm is stopped when it completes 1000 iterations. The cooling rate of SA is considered as 0.85. This rate gives the better tradeoff between the very slow cooling and fast cooling. The tabu status is controlled with tabu matrix of size *n-1* x *n* in job code system and *n* units of *tabu list* in machine code system. Tabu tenure period is the number of subsequent moves during which the last pair of solutions is to be forbidden. Tabu tenure is considered as *n* for all the systems.

4.6 NUMERICAL ILLUSTRATION

Consider the problem of 10*3: 10 jobs and 3 machines system. For both the job and machine code schemes, the initial solutions are generated randomly for all the algorithms. The objective function contains either penalty and/or priority vales, so none of the measures carries time units.

Table 4.1 Problem specification for 10*3

Job (i) no	Processing Time (t_i) time units	Due date (d_i) time units	Weight (W_i)
1	13	17	1
2	14	19	2
3	3	6	4
4	15	18	3
5	12	20	2
6	20	24	1
7	4	7	2
8	10	12	2
9	17	20	1
10	19	31	3

4.6.1 Initial solution

4.6.1.1 Simulated Annealing approach

Job code system

(a) Without priority

4 5 3 10 9 6 8 7 2 1; $f(x) = 862$

(b) With priority

2 8 5 1 10 9 3 6 7 4; $f(x) = 1866$

Machine code system

Without Priority

(a) 2 3 2 3 1 1 1 2 2 3; $f(x) = 766$

With Priority

(b) 3 2 2 3 2 2 1 1 2 3; $f(x) = 1510$

4.6.1.2 Genetic Algorithm approach

Job code system with *pop_size* = 10

86473511029 ; 43862910571; 47210395816; 45167102839; 17962534810;

76819310425 ; 52913610478; 76394102815; 43158971062; 95710318426

The best solution achievable in initial population without priority is 618.

4.6.1.3 Tabu search approach

Job code system with priority

Source string:

2 8 5 1 10 9 3 6 7 4; $f(x) = 1866.0$

Generated subneighbourhoods:

28511039674; 28511093764; 28151093674; 28519103674

82511093674; 25811093674; 28510193674; 28511093647; 28511096374

Best solution achievable in first generation = 1740

4.6.1.4 Memetic Algorithm approach

Machine code system without priority

Pop_size of 10 in each generation

2113123233; 2221212133; 1131123221; 2123321321; 1232213222

1113321212; 2312213122; 3131311221; 1222131233; 3211131311

Best solution achievable in first generation is 708.

4.6.2 Improvements in solution at generation cycles 500

4.6.2.1 Simulated Annealing algorithm approach

Job code system

Without Priority	(a) 8 1 6 3 4 5 2 7 10 9;	$f(x)$ =480
With Priority	(b) 4 3 7 8 2 6 5 9 10 1;	$f(x) = 1766$

Machine code system

Without Priority	(a) 3313112222;	$f(x) = 518$
With Priority	(b) 2233221321;	$f(x) = 1142$

4.6.2.2 Genetic Algorithm approach

Job code system without priority

38741521069; 37841521069; 38741526109; 37841526109; 78341521069;

38741526109; 37841526109; 83741526109; 37841526109 ; 38741526109

Best solution achievable in 500 generation cycles = 472.

4.6.2.3 Tabu search

Source schedule

3 2 7 8 4 5 10 1 9 6; $f(x) = 752$

Generated subneighbourhoods

23784510196; 37284510196; 32748510196; 32784105196; 32784510169;

32874510196; 32784510916; 32784511096; 32785410196

Best solution achievable in 500 generation = 752

4.6.2.4 Memetic Algorithm

Machine code system without priority

1232123331 1232123331 1232123331 1232123331 1232123331

1232123331 1232123331 1232123331 1232123331 1232123331

Best solution achievable in 500 generation = 522

4.6.3 Improvements in solution at generation cycles 1000

4.6.3.1 Simulated Annealing

Job code system

Without priority a) 87341526109; $f(x) = 472$

With priority b) 87341052196; $f(x) = 716$

Machine code system

Without priority a) 2233331112; $f(x) = 512$

With priority b) 3122332221; $f(x) = 800$

4.6.3.2 Genetic Algorithm

Job code system without priority

38741521069; 37841521069; 38741526109; 37841526109; 78341521069

38741526109; 37841526109; 83741526109; 37841526109; 38741526109

Best solution achievable in 1000 generation = 472

4.6.3.3 Tabu search

Source schedule: 2 3 7 8 4 5 10 1 9 6, $f(x) = 752$

Generated neighborhoods

32784510196 27384510196 23748510196 23784105196 23784510169

23874510196 23784510916 23784511096 23785410196

Best solution achievable in 1000 generation = 752

4.6.3.4 Memetic algorithm

Machine code system without priority

1123212333; 1123212333;1123212333;1123212333;1123212333

1123212333;1123212333;1123212333;1123212333 ;11232123331;

Best solution achievable in 1000 generation = 522

4.7 SIMULATION RESULTS OF THE APPROACHES

The four algorithms are tested with 20 different data sets by varying the number of jobs and number of machines. The machines are varied from 2 to 10, whereas the jobs are varied from 6 to 80.

MACHINE CODE SYSTEMS

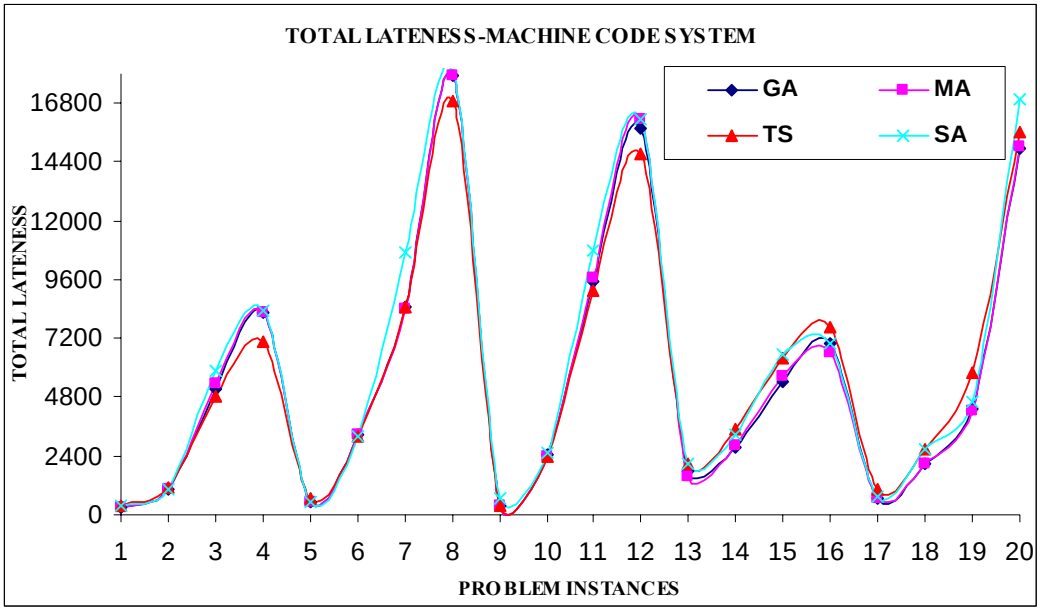

Figure 4.7 Simulation results of Total Lateness measure in machine code system

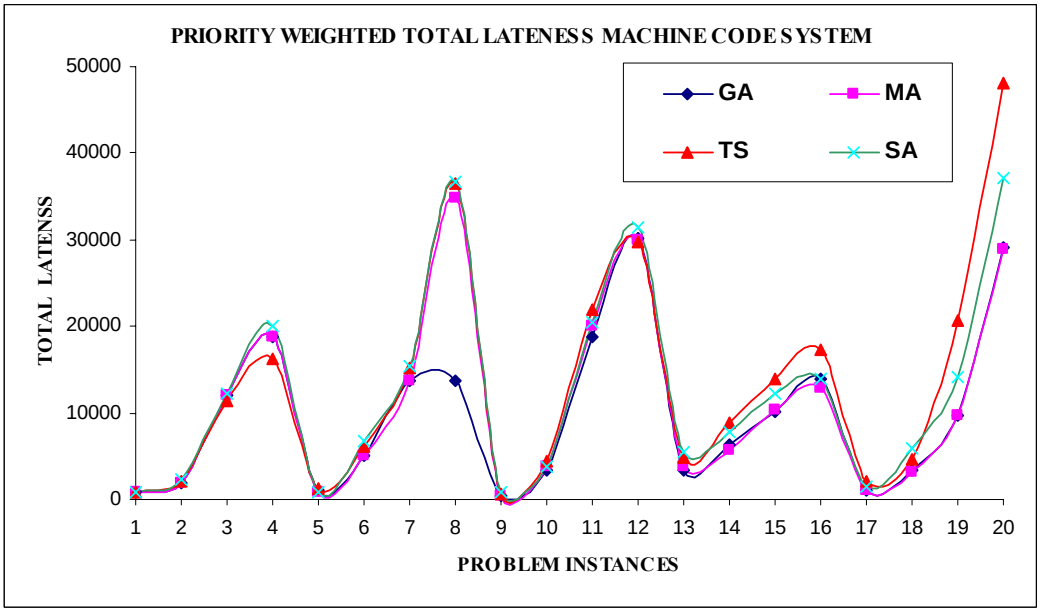

Figure 4.8 Simulation results of Priority Weighted measure in machine code system

JOB CODE SYSTEMS

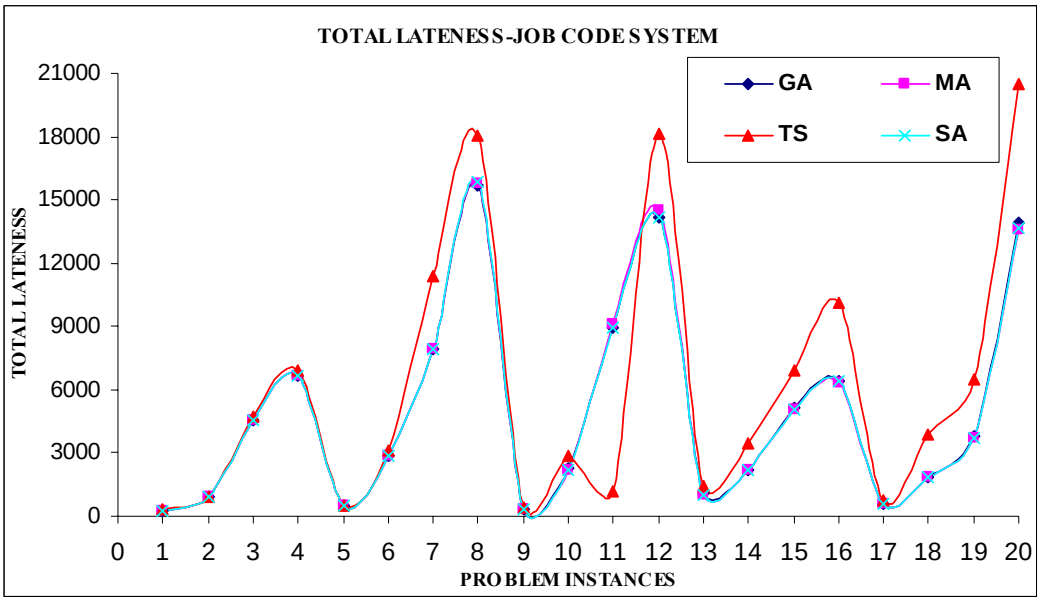

Figure 4.9 Simulation results of Total Lateness measure in job code system

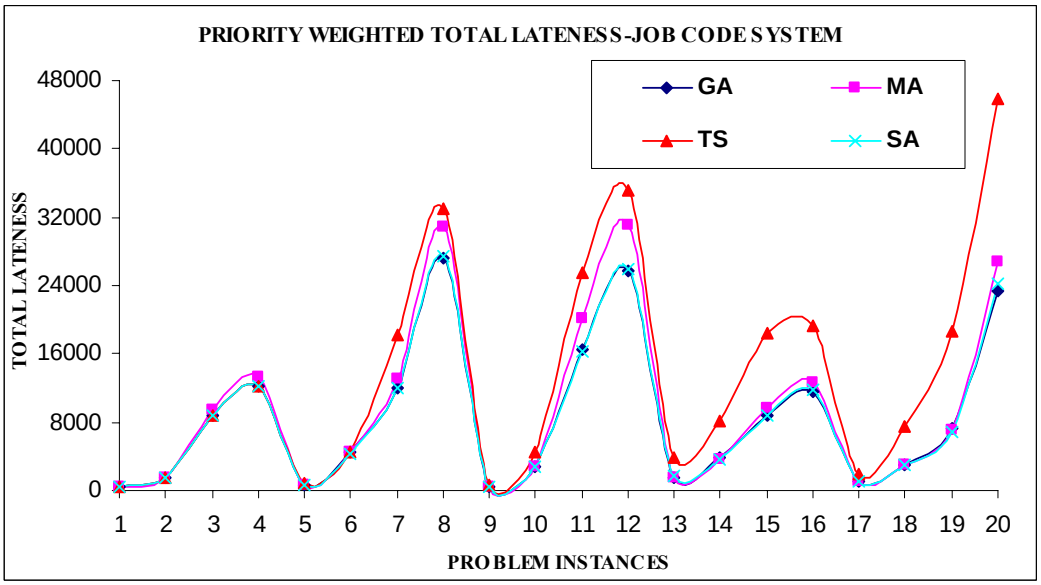

Figure 4.10 Simulation results of Priority Weighted Total Lateness in job code system

The quantitative values of the performance measures are given in Appendix B. In the Tables B.1-B.4, each field is occupied with two values [a,b], in that 'a' represents the objective function achieved in 1000 iterations and 'b' represents the computational time required to achieve the objective value.

4.8 PERFORMANCE EVALUATION

4.8.1. Lateness

The objective values achieved by various meta heuristic algorithms are represented in graphs. From Figure 4.7, it can be seen that memetic algorithm is performing better than most of the algorithms for total lateness objective function measure. It also shows that genetic algorithm is performing equally well along with memetic algorithm. But while comparing the priority weighted total lateness with machine code system in Figure 4.8, the performances of all the algorithms are seems to be equal. Figures, 4.9 and 4.10 gives the total lateness without and with priority measures respectively. For total lateness measure under job code system, almost all the algorithms are performing better than tabu search. In general, memetic algorithm is performing better under machine code system than job code system. Randomness in machine code system is giving more room for diversification while searching the minimum lateness.

4.8.2 Computational time

Duncan's multiple range tests (Montgomery, 1984) is used to compare the computational time mean of each algorithm for total lateness measure under job code system.

Hypothesis:

$$H_0 : \mu_1 = \mu_2 = \mu_3 = \mu_4$$

$$H_1 : \mu_1 \neq \mu_2 \neq \mu_3 \neq \mu_4$$

Duncan's tests for computational time mean for four algorithms are given in Table 4.2. The Duncan's test is used to determine differences among treatments at 5% level of probability.

Table 4.2 Duncan's test table

Source of variation	Sum of Squares	DOF	Mean Square
Between algorithms	63.56	3	21.18
Error (within Algorithms)	262.11	76	3.45
Total	325.68	79	

N=80, a (Levels) = 4, n=20

N = Total number of measures considered in test

a = Number of levels

n = Total number of measures considered in each levels

Ordered mean computational times

1) $\bar{Y}_{Sa} = 0.2315$; 2) $\bar{Y}_{Ts} = 1.8779$; 3) $\bar{Y}_{Ga} = 2.09$; 4) $\bar{Y}_{Ma} = 2.62$

$S_{y_i} = \sqrt{3.45/20} = 0.415$; S_{y_i} = Standard error for each mean

Consider acceptance range of 95%.

From Duncan's table,

$R_2 = r_{0.005}(2, 76) = 2.82\ (0.415) = 1.170$

$R_3 = r_{0.005}(3, 76) = 3.00\ (0.415) = 1.245$

$R_4 = r_{0.005}(4, 76) = 3.09\ (0.415) = 1.282$

Measure of difference:

4 --- 1 = 2.384 > 1.282 (significantly differ)

4 --- 2 = 0.7376 < 1.245 (no difference)

4 --- 3 = 0.525 < 1.1703 (no difference)

3 ---1 = 1.859 > 1.245 (significantly differ)

3 ---2 = 0.2122 < 1.1703 (no difference)

2 --- 1 = 1.646 > 1.1703 (significantly differ)

$\bar{Y}_{Sa}$ $\bar{Y}_{Ts}$ $\bar{Y}_{Ga}$ $\bar{Y}_{Ma}$

2.384

1.859

1.1703

Null hypothesis (H_0) is rejected for computational time mean of simulated annealing(i.e., Computational time mean achieved by simulated annealing algorithm significantly differs from other algorithms). The results indicate that the computation time achieved by simulated annealing is comparatively better than other algorithms.

4.9 INFERENCES FROM THIS CHAPTER

Various optimization techniques that are available to optimize the scheduling problems are explained. Two approaches are proposed to schedule the single stage parallel machine shops. A study is carried out to compare four meta-heuristic search methods which can be applied to solve the scheduling problem that is similar to a single stage of the mould manufacturing shop. All the methods are tested with two coding systems for a fixed number of iterations. Job based coding system is designed for structural search in solution space, whereas machine based coding system is designed for random search. Job code system of genetic algorithm is equipped with position based crossover and swap mutation, whereas machine code system is equipped with two cut point crossover and insertion mutation. Local-climb operation in memetic algorithm is done by EDD (Earliest Due Date) rule. Tabu search is designed with tabu matrix for job code system and tabu list for machine code system. Both the coding schemes in tabu search algorithm use the adjacent pair-wise interchange to create the subneighbourhoods.

Search in simulated annealing algorithm is designed with random selection from swap, lot change, and insertion operators. Computational tests were performed for 20 different set of problems. The problem instances are constructed with machines varies from 2 to 10, whereas the jobs vary from 6 to 80, so total of 320 tests is conducted to get the performance measure of total lateness with and without priority in two coding schemes. Results from studies shows that both the Memetic and Genetic algorithm gives the minimum lateness compared to tabu search and simulated annealing algorithm in most of the conditions. The computational time of each algorithm increases with an increase in problem size. The significant differences between computational time means are compared by Duncan's multiple range tests. Even though the computational time achieved by simulated annealing is considerably better than most of the algorithms, computational time itself is considered as the second choice compared to lateness measures in most of the manufacturing industries.

Studies in single stage parallel machine shops shows that population based algorithms are producing better results than single solution techniques. Machine based systems are performing well both in terms of performance measures and computational time. It may be because of the combined approach of sequencing and scheduling tasks. From literature (Glass et al., 1994 and Chang et al., 1999) and in real time systems that are available today in computer networks, the systems are scheduled by job based system rather than machine based system, so there is a chance of improvement by testing the sequencing and scheduling tasks concurrently in job code systems. Based on the above objective, a new approach is developed to combine the sequencing and scheduling tasks under job code system. The proposed approach and its working principle are presented with example in the next chapter.

CHAPTER 5

COMBINED PLANNING IN PARALLEL MACHINES

5.1 PLANNING IN SINGLE STAGE SYSTEM

Single stage production systems consist of the basic single machine problem, the extensions of the single- machine problem and the parallel machine problem. Before formulating the overall planning system of the manufacturing plant, it is advisable and useful to study the performance of each individual system, so as to have a thorough view about each stage of the plant. Most of the manufacturing plants are formed by groups of parallel machine shops with some strict job shop stages. A through study of these two systems will help one to model and predict the overall system of the plant.

In scheduling problems, it is often possible to take advantage of parallelism in resource structure. Grouping of machines based on their performance is common to all kinds of industries, so finding the optimum sequence and schedule of jobs in identical parallel machine system is one of the biggest problems for these shops. In small scale manufacturing industries, much attention is given to performance measures, whereas in parallel machine computer clusters, the focus is on computational time that is required to make scheduling decisions. A simple context for investigating the effects of parallel resources is the problem of single-stage sequencing with several machines. As in the basic model given in Figure 3.1 of chapter 3, there are n single-operation jobs simultaneously available at time zero scheduling horizons and there are m identical machines available for processing, in which a job can be processed by at most one machine at a time. Scheduling of jobs to machines and sequencing of the scheduled jobs in the machine are considered separately in most of the systems. It is obvious that models representing combined planning problems become increasingly difficult to solve as the

environment, job characteristics and objective functions become more and more realistic (Franca et al., 1994).

5.1.1 Separation of Sequencing and Scheduling in parallel machines

There are numerous studies and researches that are carried out to study the performance of the parallel machine systems. Some of the studies are focused on the improvement of regular performance measures and some have concentrated on the non regular performance measures (Moore, 1968). There are also lot of researches in optimizing sequence dependent scheduling of parallel machines and scheduling in non identical parallel machines. In almost all these studies, two important concepts of parallel machine scheduling problems are considered separately. They are 1) Scheduling of jobs to machines. 2) Sequencing of jobs in each machine. Unlike in single machine scheduling problems, these two processes are considered individually, but it is not necessary to formulate and solve them individually. In the system which considers the sequencing and scheduling separately, it is not guaranteed that optimized schedule should contain the optimally sequenced jobs. Some of the researches tried to find the optimum sequence and schedule by simultaneously solving the problem, initially for optimal schedule and subsequently for optimum sequences of the jobs (Baptiste et al., 2000). In this approach, the time required to solve the problem will increases by each task. In this chapter, a new approach is proposed to concurrently solve the sequencing and scheduling tasks in parallel machine systems.

5.1.2 New approach for combined planning

In this new approach, the sequencing and scheduling of parallel machine scheduling system is considered as a single objective and improved as one. A representation is presented to encode these two things into a single string. Each representation consists of *job lists* and *separation symbols*, in which integers are used to represent all possible

permutation of jobs and *separation symbols* (*) are used to partition of jobs to machines. A simple example with 12 jobs and 4 machines is considered here in order to explain the importance of separation symbol. Suppose the initial random selection assigns the jobs 2, 3, 4 to machine one, jobs 5,10,1 to machine two, jobs 12, 9,8,11 to machine three and jobs 6, 7 to machine four. The sequence and schedule of this system is represented as a string as follows:

{ 2 3 4 * 5 10 1 * 12 9 8 11 * 6 7 }.

In a general system of n jobs and m machines, each solution string contains n + (m-1) positions, so the solution space contains a minimum of (n + (m-1))! possible combinations. In this consideration, it is obvious that complexity grows as the problem size increases, either in terms of number of jobs or number of machines.

5.2 OPTIMIZATION OF SEQUENCING AND SCHEDULING

The problem addressed here consists of assigning n jobs to m identical parallel machines and scheduling the jobs to machines. Priority weighted total lateness measures are considered in this study. Elements of parameters in total lateness (i.e. earliness and tardiness) are considered with different penalty factors. The processing times of jobs and their due dates are generated by uniform distribution. Each job is assigned with priorities based on their importance from priority scale of 1-5. It is assumed that all jobs are ready for processing at time zero planning horizon and the machines are continuously available. Along with the performance evaluation of the new approach, the study is also used to understand the effect of variation in due date and parameters of the optimization algorithm.

The objective function of this problem is given in Eq 5.1.

$$\min_{\sigma \in \prod} \boldsymbol{f}(\boldsymbol{\sigma}) = \sum_{j=1}^{n} w_j (\alpha E_j + \beta T_j) \tag{5.1}$$

5.2.1 Memetic algorithm based system

Moscato (1989) introduced this hybrid genetic approach or Memetic algorithm, which combines the recognized strength of population based methods with intensification capability of local search. In Memetic Algorithm, all individuals of each population evolve solutions until they become local minima of a certain neighbourhood. There are two main generic operators used in this algorithm, crossover and mutation. Usually, the crossover is used as main genetic operator. These operators are modified in order to adopt the problem structure of this study.

5.2.1.1 Crossover

The existing one way position based crossover operator is modified to adopt the partition of jobs to machines and sequence of jobs to each machine. Actually, in this crossover operator, certain restrictions are incorporated in order to maintain the feasibility of offspring. Performance of the modified one way position based crossover is described below:

1) Transform all partition symbols from parent to offspring,
2) Select random positions from the same parent without any partition symbol,
3) Transform the jobs from randomly selected positions to offspring and
4) Obtain the remaining jobs from second parent by left-to-right scan.

The same process is repeated to create the second offspring by making the second parent as the base. The modified one way position based crossover with chosen positions of 5 and 9 is explained with simple representation.

Parent 1: 2 3 4 * 5 10 1 * 12 9 8 11 * 6 7

Parent 2: 7 4 1 * 2 3 6 * 11 10 9 * 5 12 8

Offspring 1: 7 4 1 * 5 2 3 * 12 6 11 10 * 9 8

Offspring 2: 3 4 5 * 2 10 1 * 11 12 9 * 8 6 7

Figure 5.1 Working scheme of one way position based crossover

Two way position based Crossover

To test the effectiveness of the crossover operator, a small modification is made in the one way position based crossover. In the two way position based crossover, the first three steps of the one way position based crossover are repeated and the final step in the crossover operation is modified to right-to-left scan instead of left-to-right scan. With the same selected crossing positions, two way position based crossover works as follows

Parent 1: 2 3 4 * 5 10 1 * 12 9 8 11 * 6 7

Parent 2: 7 4 1 * 2 3 6 * 11 10 9 * 5 12 8

Offspring 1: 8 9 10 * 5 11 6 * 12 3 2 1 * 4 7

Offspring 2: 3 4 5 * 2 10 1 * 11 12 9 * 8 6 7

Figure 5.2 Working scheme of two way position based crossover

In Figure 5.2, offspring 2 is similar to offspring 2 of one way position based crossover this because two way crossover operators only affects the second parent.

5.2.1.2 Mutation

Swap mutation is applied to make perturbation in a single chromosome. The adopted swap mutation selects two random positions and swaps elements of selected positions. The randomly swapped elements have better chance of either being a job or a partition symbol, or being both. If the selected elements contain one with job symbol and another with partition symbol, the performance of the swap mutation is considerably equal to the crossover operation. The swap mutation is explained with randomly chosen positions of 4 and 6.

Parent: 7 4 1 * 2 3 6 * 11 10 9 * 5 12 8

Offspring : 7 4 1 3 2 * 6 * 11 10 9 * 5 12 8

Figure 5.3 Working scheme of swap mutation

5.2.1.3 Local Climb Heuristic

The local climb method used in this study is **EDD** (**E**arliest **D**ue **D**ate) rule based heuristic. It is proved that maximum lateness is minimized by earliest due date dispatching rule in single machine scheduling problem (Baker, 1974). This rule is used to adjust the jobs in each sub schedule on each machine, because there is no such kind of optimality procedure that exists for parallel machine problems unlike single machine scheduling problems.

5.2.1.4 Selection mechanism

Roulette wheel with Elitism is used as the basic mechanism to reproduce the next generation based on current enlarged population. The selection scheme plays an important role in MA by improving the average quality of the population. Selection determines which individuals in the population pool will be selected for producing the next generation. This is achieved by giving higher chance to better individual chromosome or solution to be copied into the next generation based on its fitness value.

The general structure of the proposed Memetic Algorithm is described below:

K= Number of generation

L= Population size

STEP 1

K=1

Select N initial schedules $S_1, S_2, S_3 \ldots S_N$ *using* EDD heuristic rule

Evaluate each individual of the population

STEP 2

Create *N* new individuals by crossing the current population using *position crossover*

Mutate the newly created *N* individuals using *swap mutation*

STEP 3

For *L=1* to *N*

Locally climb in each schedule using EDD

Evaluate each individual of the population

STEP 4

Create the New population using selection heuristic

STEP 5

K = *K*+1

If *Stopping condition* = true

Then return the best individual as the solution and *STOP*

Else go to Step 2

Figure 5.4 Pseudo-code for modified Memetic algorithm

5.2.2 Simulated Annealing algorithm based system

Simulated annealing is a stochastic search strategy for selecting minimal configuration of the states in a system. Its basic thought is to apply randomness of the algorithm and to increase freedom of optimization of the algorithm to accept bad solution with certain probability, thus escaping from local optimum tending to global optimum. It was first proposed by Metropolis, Rosenbluth, Rosenbluth, Teller and Teller (1953). Kirkpatrick, Gelatt, and Vecchi (1983) first applied the approach successfully to optimization problems. The goal which was used is to minimize a desired objective or energy function. Starting from an initial solution, the algorithm unconditionally accepts the solution which results in smaller energy values than the last solution. A new solution with larger energy

value is accepted or rejected based on the value of a probability function. The ability to occasionally accept degenerate solutions is what separates simulated annealing algorithm from other gradient-descent methods. In gradient descent methods, once a local minimum of the objective function is reached, there can be no further improvements. Simulated annealing has been proven to converge asymptotically to global minimum (Lundy & Mees, 1986), as the number of iterations goes to infinity. However, it may also find near-optimal solutions in relatively fewer iterations. There are two main operators which performs most functions in simulated annealing algorithm. They are control parameter and mutation operator. Simulated annealing involves a control parameter that is referred to as temperature. The temperature decreases during each iteration, which in turn affects the acceptance of new state. As the temperature decreases, it is likely to accept a degenerate state. At low temperature, the algorithm approaches a gradient-descent method. The cooling parameter which is used in this approach, will give better trade off between very fast cooling and slow cooling.

$$T_{t+1} = C * T_t \tag{5.2}$$

Where C is constant, which can be varied from 0 to 1. After some random trials of experimentation to avoid the premature convergence of solution space and escape from local optimum, the cooling parameter is fixed to 0.88. In order to improve the performance of simulated annealing algorithm, A new method which combines two techniques of perturbation is introduced. Two perturbation techniques that are used are the swap method and inversion method. During each generation, any one of these method is selected randomly to perturb the instance of solution space.

Generate *random integer Z*

(Z = 0 or Z = 1)

If *Z= 0*, then perform mutation

Inversion based perturbation

If *Z= 1*, then perform mutation

Swap based perturbation

Figure 5.5 Pseudo-code for new mutation technique

Inversion method

Consider the same example problem which is mentioned above with the initial solution.

Initial solution = { 2 3 4 * 5 10 1 * 12 9 8 11 * 6 7 }

Consider the inversion positions of 3 and 6.

Modified solution = { 2 3 10 5 * 4 1 * 12 9 8 11 * 6 7 }

Swap method

Initial solution = { 2 3 4 * 5 10 1 * 12 9 8 11 * 6 7 }

Consider the swap positions of 3 and 6.

Modified solution = { 2 3 10 * 5 4 1 * 12 9 8 11 * 6 7 }

The implemented simulated annealing algorithm with local climb is presented here.

STEP 1

Randomly generate a feasible schedule called *Current schedule*

STEP 2

Start from the initial temperature $T=T_0$, while not reaching the final temperature T_{lowest}

{

Generate a *random number Z*

(Z = 0 or Z = 1)

If *Z = 0*, then change the *current schedule* by inversion method *(or)*

If *Z = 1*, then change the *current schedule* by swap method

a. Make a random change to the *current schedule*,

Let *temp schedule* be the schedule after change.

b. Arrange the jobs in sub schedules by *EDD* rule

c. Check to make sure that *current schedule* is valid. Otherwise, go back to a)

d. Calculate the objective value of *current schedule* (F_1) and *temp schedule* (F_2).

If ($F_2 < F_1$)

Let *temp schedule* be *current schedule*

Else

Randomly generate X (0<X<1)

If $X < e^{(Delta)/T}$; Delta = F_1-F_2

Let *temp schedule* be *current schedule*

Else

Let *current schedule* remain unchanged

End if

End if

e. Repeat (a) to (d) until a criterion is satisfied.

f. Reduce the temperature to new T

}

Figure 5.6 Pseudo-code for Simulated Annealing algorithm

5.3 PERFORMANCE OF THE APPROACHES

An experiment is conducted to find the performance of the proposed approach in combined planning. The processing time and priorities of jobs are generated by uniform distribution in ranges of 1 to 20 and 1 to 5. Two different due date schemes considered in this study. Based on distinct due date criteria, due date is calculated by the product of due date factor and processing time of the job.

$$\left| duedate_j \right| = \text{Processing time}_i * \text{due date factor} \qquad (5.3)$$

Details about the jobs are given in Appendix C. Two different due date factors, 1.3 and 1.4 are considered in order to study the effect of tightness of the deadlines. It is obvious that considering 130% due date factor will provide tighter schedule than 140%. Based on random simulation trials, it was found that due date factor less than 1.3 will result in highly disturbed schedules which are hardly possible to implement. On the other hand, considering the due date factor more than 1.4 will provide very loose schedules and optimization will become unnecessary in that case.

For comparison purpose, both the algorithms are simulated for 1000 iterations. Based on random studies, crossover and mutation probability of memetic algorithm is fixed as 85% and 60% respectively. Population size is of 10 is considered in memetic algorithm. Penalty values of 4 and 2 are applied to tardy and early jobs in both the approaches. The initial and final temperatures of simulated annealing algorithm are assigned as 6500 and 0.0010 with cooling rate of 0.95. Even though the assigned range of temperature generates more than 1000 iterations, simulation is stopped after 1000 iterations for comparison purpose. The simulation results are presented in Figures 5.7-5.10.

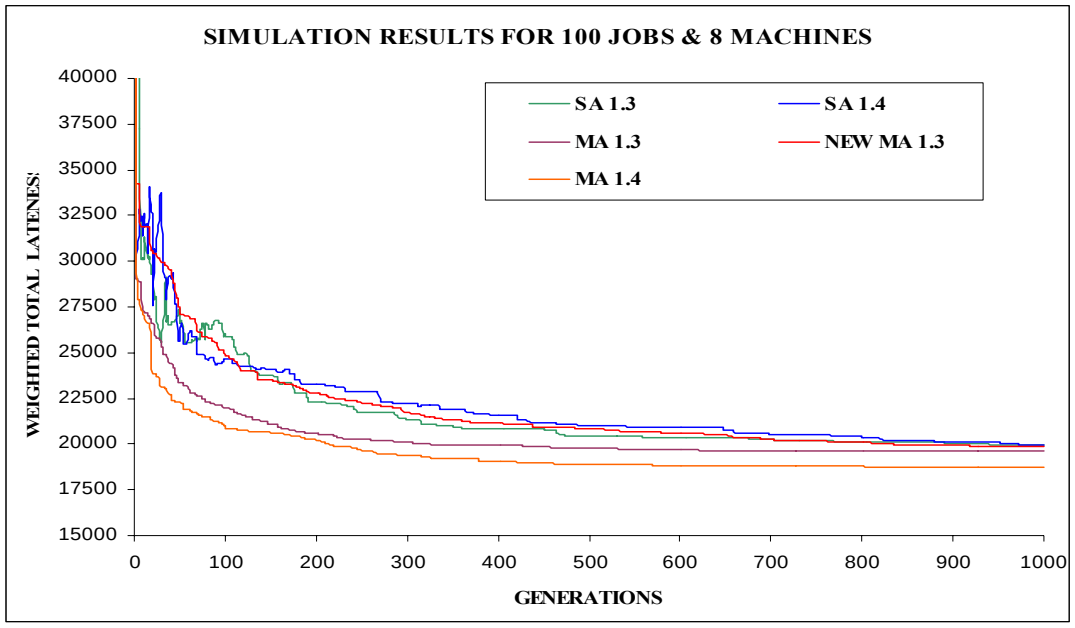

Figure 5.7. Simulation results for 100 jobs and 8 machines case

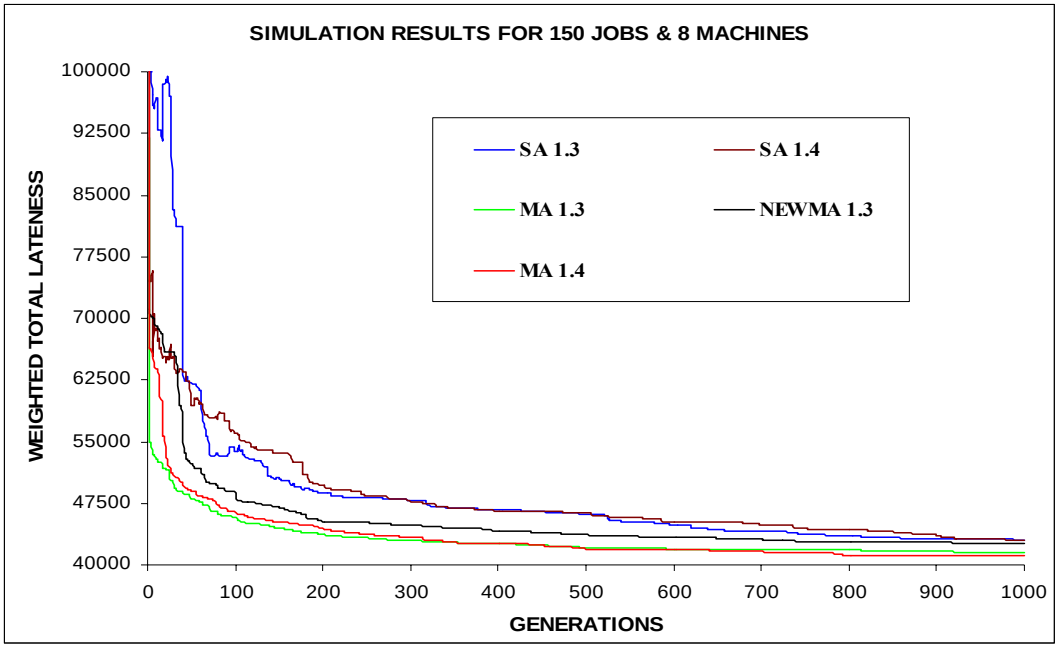

Figure 5.8 Simulation results for 150 jobs and 8 machines case

SA 1.4 = Simulated Annealing algorithm with 140% due date factor

SA 1.3 = Simulated Annealing algorithm with 130% due date factor

MA 1.4 & MA 1.3 = Memetic Algorithm with 140% and 130% due date factor

NEW MA 1.3 = Modified crossover in Memetic algorithm with 130% due date factor

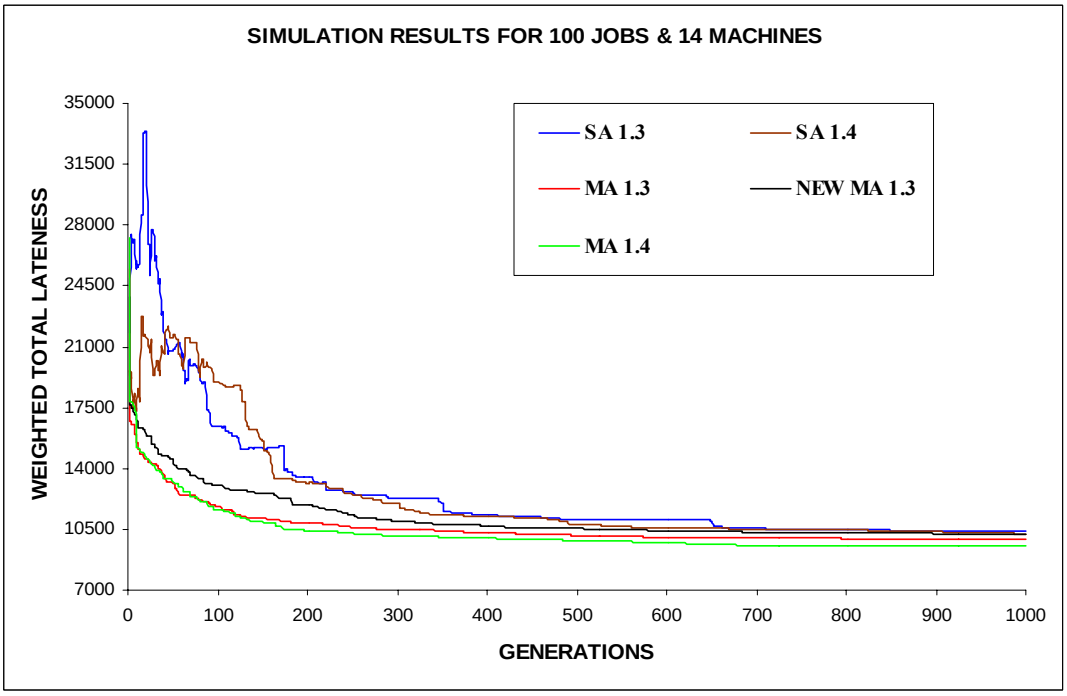

Figure 5.9 Simulation results for 100 jobs and 14 machines case

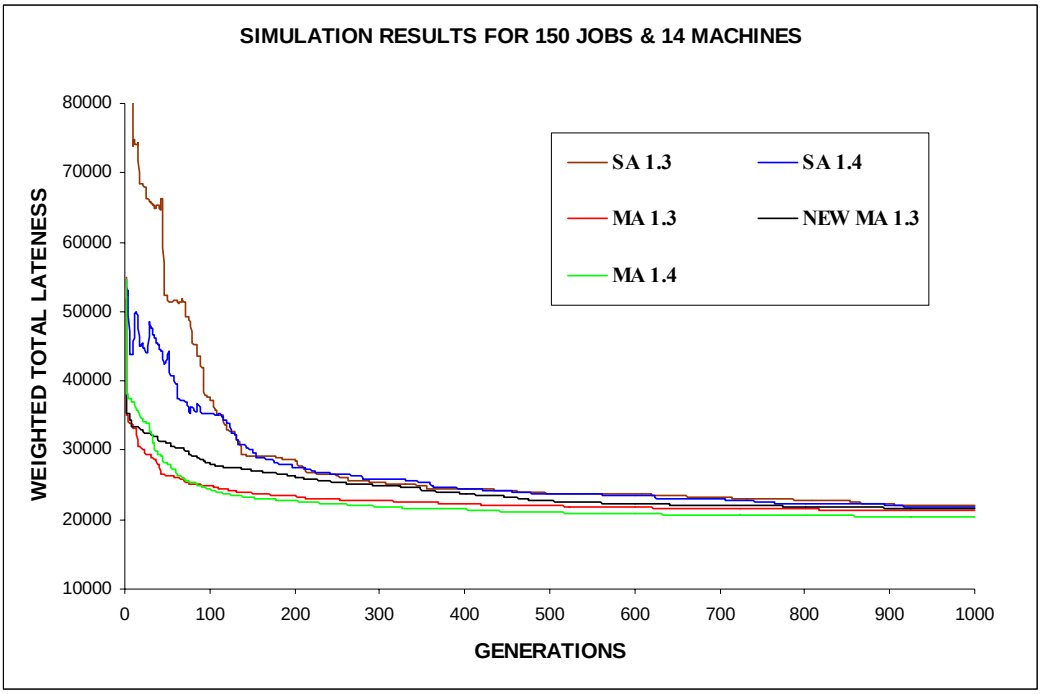

Figure 5.10 Simulation results for 150 jobs and 14 machines case

Performance of the five different approaches is presented in Figures 5.7 to 5.10. NEW MA 1.3 is considered to study the effect of two way position based crossover operator in memetic algorithm against one way position based crossover operator. From the results, it is proved that memetic algorithm with 1.4 due date factor is considerably producing better solutions in parallel machine systems. Results show that one way position based crossover operator is working better in almost all cases than two way position based crossover operator. Even though the initial perturbation of two way position based crossover is high, it converges quickly with less optimal solution than one way operator. Chosen number of generation cycles is proved to be sufficient for this experiment by convergence of the approaches. Figures 5.8-5.10 show that NEW MA 1.3 is performing better than SA 1.3 and 1.4. One of the possible reasons for this improvement is either because of problem size and better perturbation during the simulation experiment. In general, Memetic algorithm is working better than simulated annealing for solving combined planning tasks in parallel machine system.

5.4 INFERENCES FROM THIS CHAPTER

In this chapter, studies are carried out to solve the combined sequencing and scheduling problems in parallel machine systems. Two algorithms namely simulated annealing and memetic algorithm approaches are used for study. Two different due date factors are considered in study and their effects of this variation is studied against performance measures. Priority weighted total lateness is considered as the performance measure. Penalty value of 4 is assigned to each tardy job, while penalty value of 2 is assigned to each early job. Compare with other optimization studies, large problem sizes are considered which ranges from 100, 150 jobs and 8, 14 machine combinations. Effect of changes in crossover operator of memetic algorithm is also studied in the experimentation. Simulation results shows that the use of memetic algorithm to minimize priority weighted total lateness on parallel machines has produced better results compare to simulated annealing algorithm. Memetic algorithm with 140% due date factor yielded good results for most of the problems. Simulation results also represents that better performance of MA occurs when due date factor is 1.4 than 1.3. From the simulation results, memetic algorithm with 130% due date factor claims to be superior to simulated annealing with 140% due date factor.

In terms of computational time, simulated annealing algorithm works better than memetic algorithm. But in manufacturing industries, priorities will be given to performance measure rather than computational time. From results, it is clear that one way position based crossover in memetic algorithm is performing better than two way position based crossover. In overall, memetic algorithm with position based crossover operator is performing better than simulated annealing algorithm for combined sequencing and scheduling of jobs in single stage identical parallel machine systems.

CHAPTER 6

SCHEDULING FLEXIBLE PROCESS PLANS

Scheduling and Process planning are two of the important planning tasks in manufacturing industries. Numerous researches have carried out to integrate these two planning activities. Flexibilities in the manufacturing environment increase the chance of integrating planning activities. Flexibilities exist in process plans in the forms of alternative machines to manufacture the same part. Number of identical machines in the manufacturing environment increases the flexibilities further. Alternative machines to process the same part help to generate different process sequences for that part. Along with flexibilities, constraints are available in manufacturing systems, which have to be satisfied in order to achieve the feasible solution. These opportunities in manufacturing environment helps the planners to generate flexible process plans in order to execute the planning activities smoothly.

6.1 PROBLEM STATEMENT

The following studies are performed to tackle the scheduling problem of a mould manufacturing shop. In general, mould manufacturing shop is a complex domain which neither fits into semi automated shops nor into fully automated shops. Thus, the mould shop scheduling is complex problem as it has multiple parallel machines in each department. Multiple operation sequence for each job also increases the combinatorial search space in the mould shop scheduling problems. In most of the mould manufacturing industries, identical machines are grouped in to departments. The part has to pass through these departments in order to complete all its operations. Problem arise while scheduling the jobs in each department and moving the jobs to different departments. Number of machines in each department frames the solution space for

optimization problem. However, prevalent constraints between the operations reduce the solution space considerably. Constraints also serve the purpose of maintaining the feasibility of the generated solution. In order to control the planning activities in mould manufacturing shops, existing flexibilities and constraints have to be managed effectively and efficiently. In this study, scheduling problems prevalent in mould manufacturing shop is explored.

6.2 SCHEDULING FUNCTION

The objective of scheduling is to achieve a schedule for all job's operations onto all available machines such that a certain optimization criteria or performance measure is improved. The choice of the performance measure is critical in deciding the outcome of the schedule. Performance measures such as makespan, flow time, lateness with distinct penalty for early and tardy jobs are considered for study. The impact of priority to job is also tested. Priority addition is important criteria in scheduling problems. Priority indicates the importance of job. Usually priority is fixed with scale, lower in scale represent the less important jobs, whereas higher in scale represents very important jobs. Priority addition only affects the due date based performance measures. Multiple objective optimizations are another important concept in current optimization community. In multiple objective optimization problems various measures are improved at a time. Multiple objective optimizations concurrently improve the various performance measures. The following conditions are assumed in the scheduling function of the system:

1. Each machine can process only one operation at a time.
2. Each machine is continuously available for production.
3. An operation may not begin until its predecessors are complete.
4. Jobs consist of any number of operations.

5. Preemption is not allowed in the system.
6. Processing times and due dates are assigned while executing the system.
7. Setup time and other planning times are included in the processing time.

The approach is developed to generate feasible schedules with these assumptions and flexibilities. Flexibilities and constraints exist in mould shops are further elaborated in the subsequent sections.

6.3 PROCESS PLANNING

Process planning is another important task which helps to control the planning activities in manufacturing industries. Efficiently designed process planning system is needed in order to integrate the planning activities. Even small problems in process planning system will make the other planning activities as impossible to follow. In order to improve the scheduling functions, process planning systems has to provide as many number of flexible process plans as possible. In order to create better process planning system, the following condition has to be satisfied (Gan P Y et al., 2002):

a) The operations selected must be able to produce the desired geometrical features on the part.
b) Selected machining departments or machines must be able to perform the required machining operation with suitable tools and power.
c) Precedence constraints must be set to rule out illogical sequences

6.3.1 Flexibility in machines

Flexibility in machines is the important criteria to improve the scheduling function of flexible process plans in the mould manufacturing shops. Machine flexibility is defined as the choice of machines to perform the operation rather than dictation of single specific

machine. Flexibility in machines is accompanied by two ways. One is the flexibility adopted by operation of the job, which can be done by number of machines. Another kind of machine flexibility is the availability of number of identical machine.

Consider a single cluster parallel machine shop which contains 10 machines, and one operation has to be done by any one of the machine on that cluster. There are 10 possibilities to finish that operation. Each individual stage of the mould shop is represented as a parallel machine cluster either identical or variable machines. Even though the machine flexibility increases the chance of achieving the optimal solution, it also increases the solution space.

6.3.2 Precedence relations between operations

Efficiently designed process plan provides the precedence relation between the operations. In general precedence constraints exist in two forms, namely precedence constraints between jobs and precedence constraints between the operations. Precedence relation between the operations can be clearly explained with roughing and finishing. In general, roughing precedes finishing in manufacturing of most part. Precedence relations between the operations form the tree structure in solution space, which makes the searching to be complex. However, precedence constraints reduce the solution space in large scale systems.

6.3.3 Precedence relations between jobs

Precedence relations between jobs are the most frequent one in assembly shops. Predecessor job entirely controls all the operations of the successor jobs. For example in assembly shops, the subassemblies should be processed first, before the final assembly, otherwise bottleneck condition will occur in assembly shop (Kim and Egbelu, 1999). In mould shops, this situation occurs when considering cases where special jigs and fixtures

for a particular CNC operation. The jigs can be taken as the predecessor job for the CNC operation. Precedence constrains between the jobs are also exists in the case of making electrodes for the EDM operation (Wang and Li, 1987), so while scheduling the jobs and operations in mould shop , the precedence constraints should be considered in order to maintain the feasibility of the schedule.

As mentioned above, a mould shop is a perfect environment to test the scheduling approaches of flexible process plans. To solve the scheduling problem in mould manufacturing shop without violating the precedence constraints between the jobs and operations, a new approach is developed. A prototype model of the mould manufacturing environment is tested by genetic algorithm and simulated annealing algorithm.

6.4 SOLUTION SPACE

In optimization, solution space is one of the main criteria which decides the searching time and possibility of achieving the global optimum. It is a known fact that increases in solution space directly proportional to searching time of the optimization technique. The solution space for simplified mould shop system is explained with the example.

Table 6.1 Details of jobs in a simplified mould shop

Jobs/ Operation	Cluster A		Cluster B		Cluster C	
	M 1	M 2	M 3	M 4	M 5	M 6
1 / 1	4					
1 / 2			5	5		
1 / 3					6	6
2 / 1			3			
2 / 2	7	7				
2 / 3					10	10
3 / 1			2	2		
3 / 2					4	4
3 / 3		5				

If the search space is very large, it is advised by optimization theory to find the potential regions of optimal solutions first and directing the search towards the potential region. Table 6.1 gives the simplified representation of mould shop. Three jobs with each job consists of three operations are considered in the simplified mould shop. In order to calculate possible process plans in the system, first consider the possible process plans for job 1. Operation 1 of job 1 has to be processed by first machine of cluster A. Operation 2 of job1 can be processed by either of two machines in cluster B. In the same way, operation 3 of job 1 can be processed by either of two machines in cluster C. Assume, that there is no precedent relations between the operations of job 1, there are 3! possible ways to arrange the operations. With the inclusions of parallel machines, there are 3! × (2)(2) possible plans available to job 1. Similar to this, there are 24 possible process plans available for job 2 and 3! × (2) (2) possible process plans available for job 3. In total, there are 72 possible process plans available in simplified mould shop system. In general, the number of possible process plans P_i for job i with n number of operations in each job is,

$$P_i = n! \times Œ_{\text{for all n}} (\text{number of parallel machines})_j \qquad (6.1)$$

With the addition of constraints, the problem will become much more complex. However, inclusion of precedence constraints helps to reduce the solution space considerably. In the above mentioned problem, addition of 1 precedence constraint between the operations 1 and 2 of job 1 will reduce the possible process plans for job 1 to half than previous possibilities. Inclusion of subsequent precedence constraints reduces the solution space further. Complexity of the searching scheme increases with number of constraints and operations.

6.5 REPRESENTATION OF SOLUTION

In mould manufacturing shop, each job has single or a series of operations. Most of the mould manufacturing shops are constructed with series of parallel machine clusters. The operation of each job can be either performed by single machine or alternative machines. Most of the operations can be performed in any one of the machine in the particular cluster. Sometimes, the choice of the machine for operation is restricted by size and shape of the machine and job. Therefore, a solution string of the schedule should contain the information of selected machine standing for each operation. In general, the solution representation is defined as a series of positive integer numbers representing the operations for the entire jobs in the system. The representation scheme of solution string in mould shop scheduling is shown as follows.

Job 1 **Job 2** **Job 3** **Job n**

| ***5 2 3 4 nt 1 1 2 3 1 5 ….nt 2 4 1 5 ….nt 1 …….1 2 3 4 8……***|

nt indicates the possible number of machines available for processing the operations.

In this solution representation, the cardinality of elements for each part means the number of operations and the assigned values to the elements means the machine type number for machining the operations. The assignment of machine types to operations is done by randomly choosing the machine number from possible machines that can process the particular operation. The initial solution string is randomly selected from the range **| 1, *nt* |.** In the representation scheme, ***nt*** represents the possible number of machines available for processing the particular operation and **n** represents the number of jobs in the system.

For example consider the scheduling of three jobs, with job 1 contains 2 operations, job 2 contains 3 operations and job 3 contains 2 operations, so the solution space will contain 7 elements which represent the machines that will process particular operations.

| 5 7 10 1 3 1 2 3 |

Where | 5 7 | stands for the process plan for job1, | 10 1 3 | stands for the process plan of job 2, and | 1 2 3 | stands for the process plan of part 3.

Based on the above mentioned approach, the prototype mould shop is modeled and solved by meta heuristic algorithms. The prototype system is assumed with three clusters of parallel machines available for processing. There are 20 jobs with several numbers of operations for job is considered to be available in the system. The details of operations and its flexibility in different machines are given in Appendix D.

6.6 GENETIC ALGORITHM BASED SYSTEM

As described in chapter 4, Genetic algorithm closely follows evolution theory. Genetic algorithm strictly follows the survival of the fittest principle. Fittest solutions in a population get progress while inferior solutions are slowly moves out over generations.

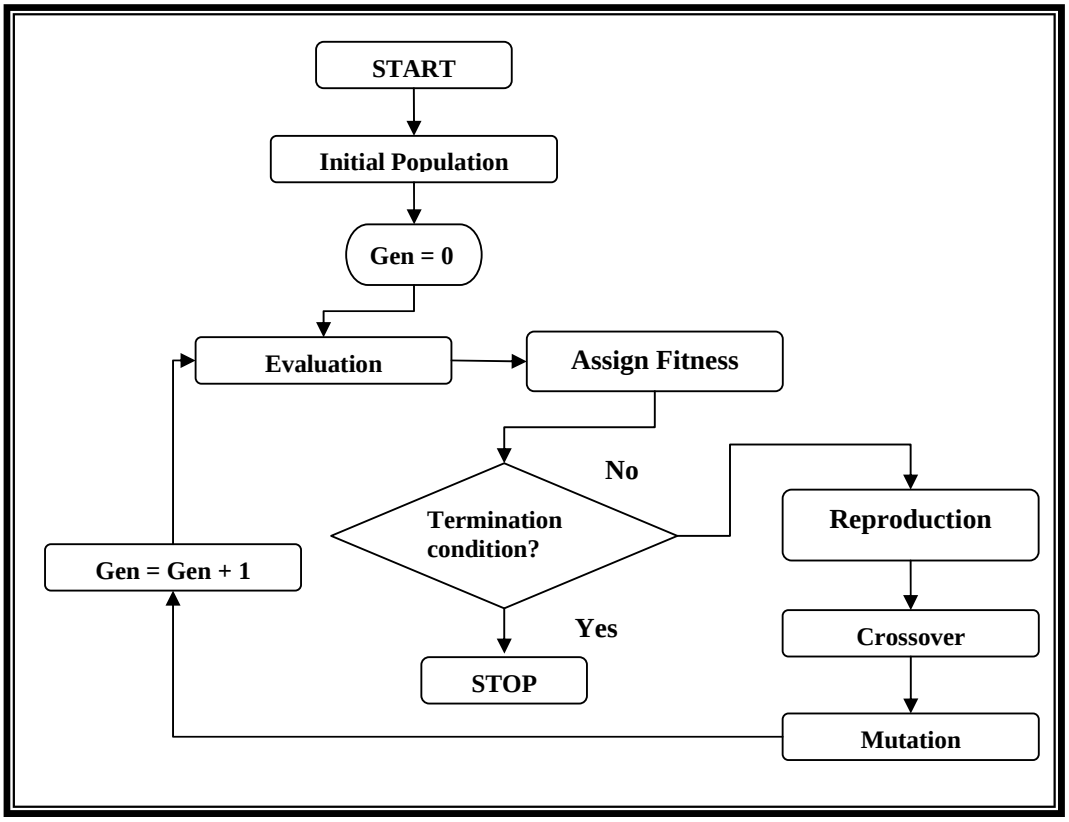

Figure 6.1 Flowchart for the implemented Genetic algorithm

The flow chart of the genetic algorithm which is used in this study is represented in Figure 6.1. The following parameters are used in genetic algorithm, while testing the prototype mould shop system. The population size of 10, crossover probability of 0.85 and mutation probability of 0.80 is used in studies. Two point crossover operator is used for crossover function, while mutation is performed by swap mutation. The detailed explanation about the functions of two point crossover operator and swap mutation is given in Figures 4.5 and 4.6 of chapter 4. The selection operation is done by elitism with insertion functions. The generation number is selected based on the size of solution space. In the prototype mould shop system studies, the genetic algorithm based system is tested for two different number of generation cycles.

6.7 SIMULATED ANNEALING ALGORITHM BASED SYSTEM

In early 80s, simulated annealing algorithm was introduced to the combinatorial optimization community. For minimization problems, it attempts to avoid the solutions being trapped into the local minima, while searching for the global minima. SA gets its name from the physical annealing of solids-heating a solid to a very high temperature and then cooling it at a slow rate, spending a relatively large amount of time near the freezing point of the solid. When this heating and subsequent slow cooling occurs, the particles within the solid rearrange themselves. The purpose of annealing is for the particles to arrange themselves in such a way that the solid posses some desired attributes, such as high strength and surface hardness. The detailed description about the simulated annealing algorithm is given in chapter 4.

The general structure and working principle of the implemented simulated annealing algorithm is presented in flowchart in Figure 6.2. The acceptance probability and cooling scheme are two of the main parameters which decide the control of the simulated

annealing algorithm. The acceptance probability decreases as temperature decreases. Mostly used acceptance function is the Boltzmann's expression.

$$P = e^{-\Delta F / T} \tag{6.2}$$

Where, ΔF = F (Schedule $_{New}$) – F (Schedule $_{Old}$)

T is the temperature and F is the objective function of the system calculated at schedules new and old. For larger ΔF, the probability of acceptance diminishes, and when ΔF is non-positive the new schedules are always accepted. As temperature decreases, the probability of acceptance decreases. In order to reduce the temperature, there are many different functional forms are reported in the literature.

The cooling schedule used in the study is given below:

$$T = \frac{T_o}{(1+t)} + T_\theta \tag{6.3}$$

Where t is the number of iterations so far, and T_θ is the lowest temperature value. Initial temperature T_0 should be high enough in order to ensure the acceptance of first few solutions in the system. The above mentioned cooling schedule is suggested by Szu (1986), which is known as one of the fast cooling schedule. For comparison purpose fixed number of generation cycles is used in the study. Following parameters are used in the simulated annealing algorithm, while testing the prototype mould shop system. Initial and final temperature of the algorithm is fixed as 5000 and 0.10 respectively. However, tests are conducted based on the fixed number of generation cycles than temperatures. Generation cycles 500 and 1000 are used in the simulated annealing approach in order to compare the effect of generation cycles.

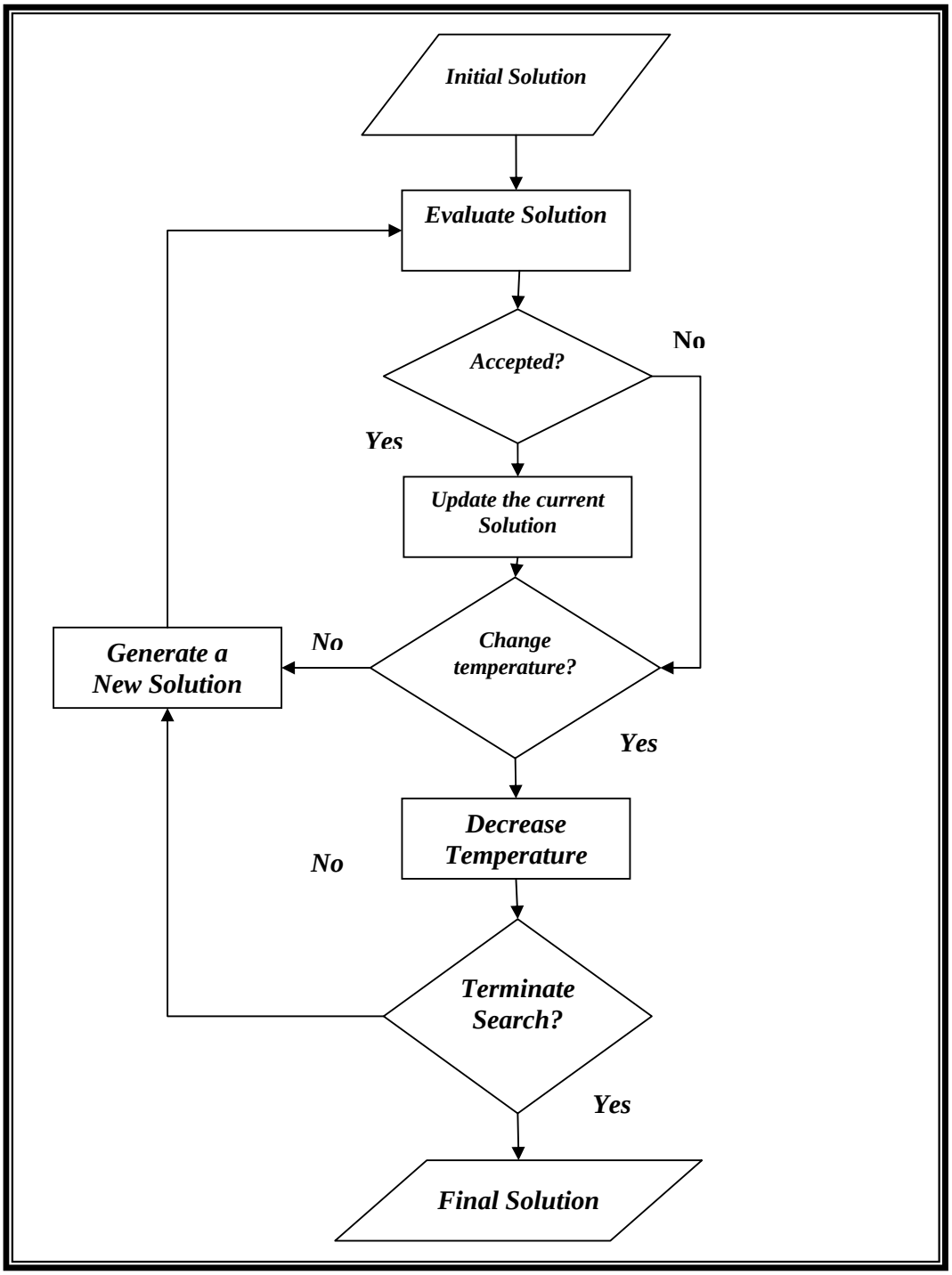

Figure 6.2 Flowchart for the implemented Simulated Annealing algorithm

Both of these approaches (simulated annealing and genetic algorithm) are studied under two different conditions in the prototype mould shop system. One without the consideration of priority in the systems, and the another one with the consideration of priority weighted measures in the system.

6.8 CASE STUDY I

The problem environment in this case: n=25, Nc= 3; $MNc_X = 3$, $MNc_Y = 3$, $MNc_Z = 4$.

Details about the jobs used in prototype mould shop system are given in Appendix D. In Table D.1, the first column represents the job number, whereas the second column gives the due date for each job. The third column represents the operations in each job. The last two columns represent the precedent constraints between the jobs and precedent constraints between the operations respectively. Precedent constraints between the operations can be better explained with 3rd operation of job 3. Operation 3 of job 3 is constrained by operations 1 and 2 of the same job, so in order to process the operation 3, operations 1 and 2 has to be completed first. Precedent constrain between the job can be better explained with job 23. In order to start the processing of job 23, all the operations in job 15 have to be completed. Precedent constraints in the system should be followed strictly to produce feasible schedule. The achieved schedule of the process plans without priority for minimizing the makespan of the overall system using simulated annealing algorithm is presented below:

Execution Time: 0.030 Seconds; *Makespan of the Entire System*: 50 time units

Schedule for the Achieved Makespan

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

The Gantt chart for this schedule is given in Figure 6.3. The makespan related objectives gives the indication about the utilization of machines in the system. The precedent constraint of jobs and operations plays an important role in makespan related objectives.

Cluster X contains 4 CNC Machines; Cluster Y Contains 3 Wire Cut Machines, and Cluster Z contains 3 EDM machines.

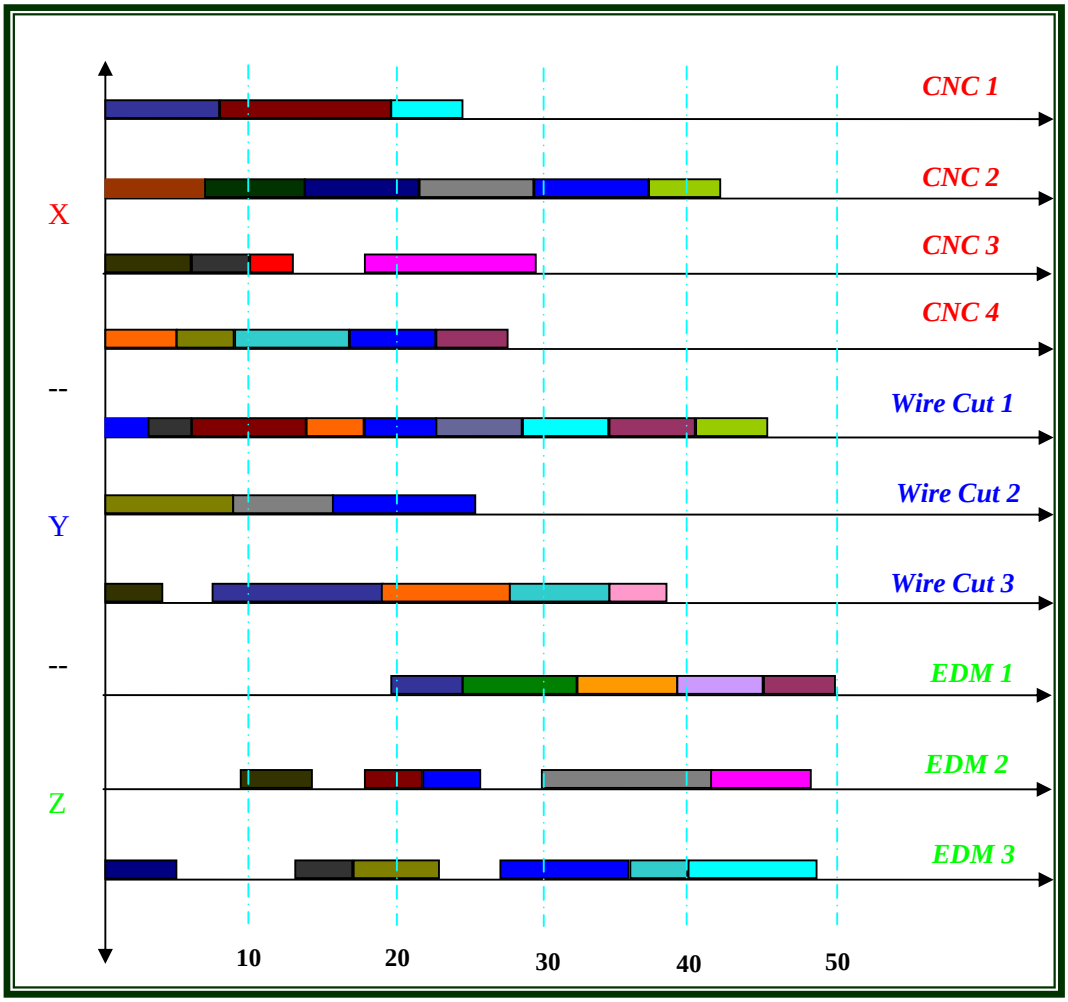

Figure 6.3 Gantt chart for schedule obtained by SA based system for case study I

In the above Gantt chart, ideal time occurred either because of the precedence relations between the jobs or precedence relations between the operations. It is difficult to reduce this ideal time from the system. The overall makespan achieved by simulated annealing based approach is 50 time units. From Gantt chart, it is clear that utilization of machines in cluster X and Cluster Y are considerably better than Cluster Z. All the machines in cluster Z have some ideal time. The optimization algorithms are used to improve the overall system, so search is directed to find the better schedule with the consideration of three clusters at the same time.

Performance of SA based system in minimizing various measures for the above given case is presented graphically as;

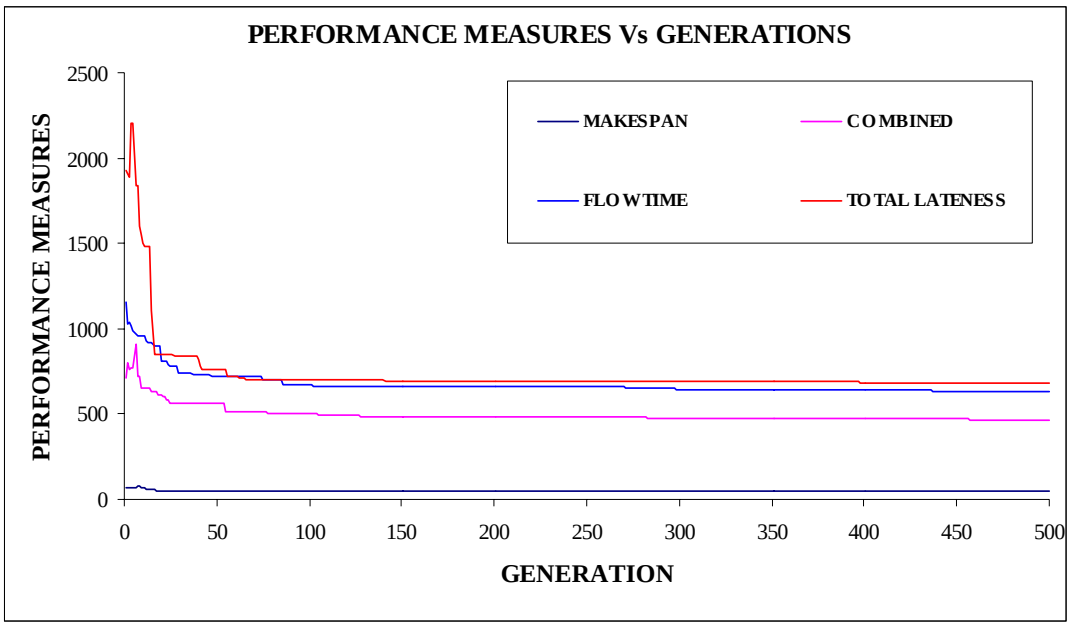

Figure 6.4 Performance of SA based system for Case Study I

The performance of the genetic algorithm based system for this case is given as below:

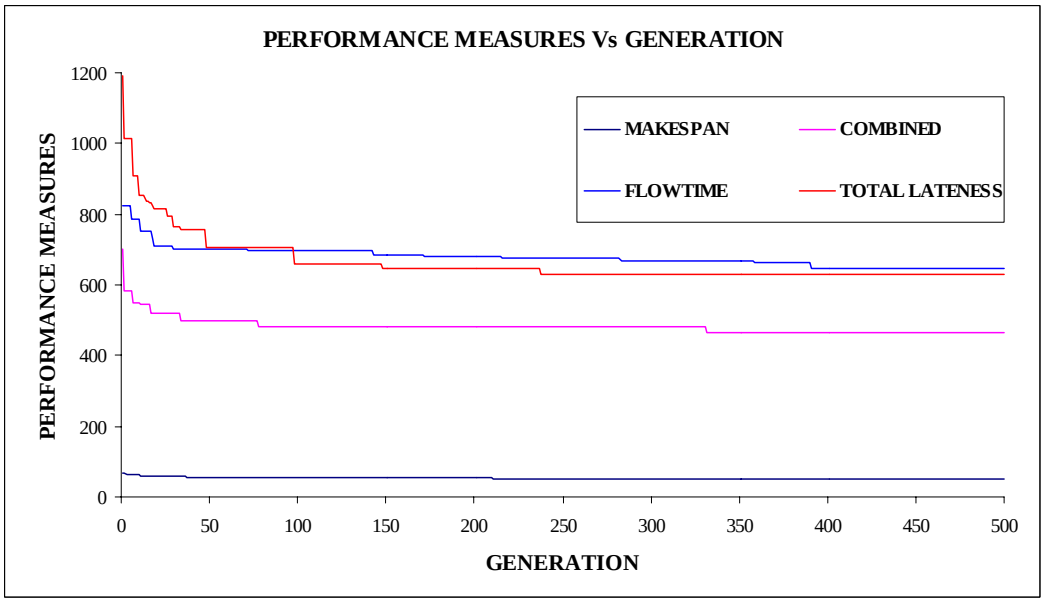

Figure 6.5 Performance of GA based system for Case Study I

Some of the Best Schedules of process plans by SA based system

By the consideration of Makespan as objective function

Execution Time: 0.050 Seconds; Minimum Makespan Achieved: 49

Schedule of process plans for achieved Makespan

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

By the Consideration of Flow time

Execution Time: 0.030 Seconds; Minimum Total Flow time achieved: 642.0000

Schedule for achieved minimum total flow time

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

By the Consideration of Multiple Objectives

Execution Time: 0.040 Seconds; Minimum Combined objective: 500.8000

Schedules for obtained minimum combined objective function

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

Some of the best Schedules of process plans by GA based system

By the consideration of Total lateness with tardy penalty 3.0 and early penalty 2.0

Execution Time: 0.070 Seconds; Minimum achieved Total Lateness: 620.00

One of the best schedules and Total lateness from final population

6	3	3	7	10	3	3	9	1	5	8	4
5	10	1	6	10	3	5	3	7	9	9	7
10	6	2	7	9	4	5	9	6	10	4	7
9	1	5	9	8	4	3	7	10	2	1	4
6	8	6	*Total Lateness: 620.00*								

By the consideration Of Makespan,

Execution Time: 0.090 Seconds; Minimum Makespan: 48.00

6	3	2	7	10	3	3	8	1	7	9	4
6	8	2	6	9	3	6	4	6	10	10	6
9	5	1	5	8	2	5	10	7	8	3	7
9	2	5	10	10	1	4	5	9	2	1	4
6	8	6	*Makespan: 48.00*								

From these studies, it is clear that performance of the genetic algorithm is considerably better for all most all the measures. This result justifies the population based optimization techniques are performing slightly better than single solution based techniques for scheduling problems.

6.9 CASE STUDY II

The problem environment in this case: n=25, Nc: 3; $MNc_X = 4$, $MNc_Y = 3$, $MNc_Z = 3$.

Details about the jobs used in prototype mould shop system are given in appendix E. In this study the problem environment which is mentioned in the previous case study is considered with priority tagged to each job to indicate the importance of the job. The

second column in the Table E.1 contains the priority vale of each job. As mentioned before, priority values are important to indicate the importance of their individuality in the system. In both the cases, whenever the lateness occurs the early jobs are penalized by the penalty value 2.0 and the tardy jobs are penalized by penalty value 3.0. Both Genetic algorithm and Simulated Annealing based systems are iterated for fixed number of generation cycles. The best performance of the SA based system by minimizing the Total Lateness in 1000 iterations is given in Gantt chart 6.6.

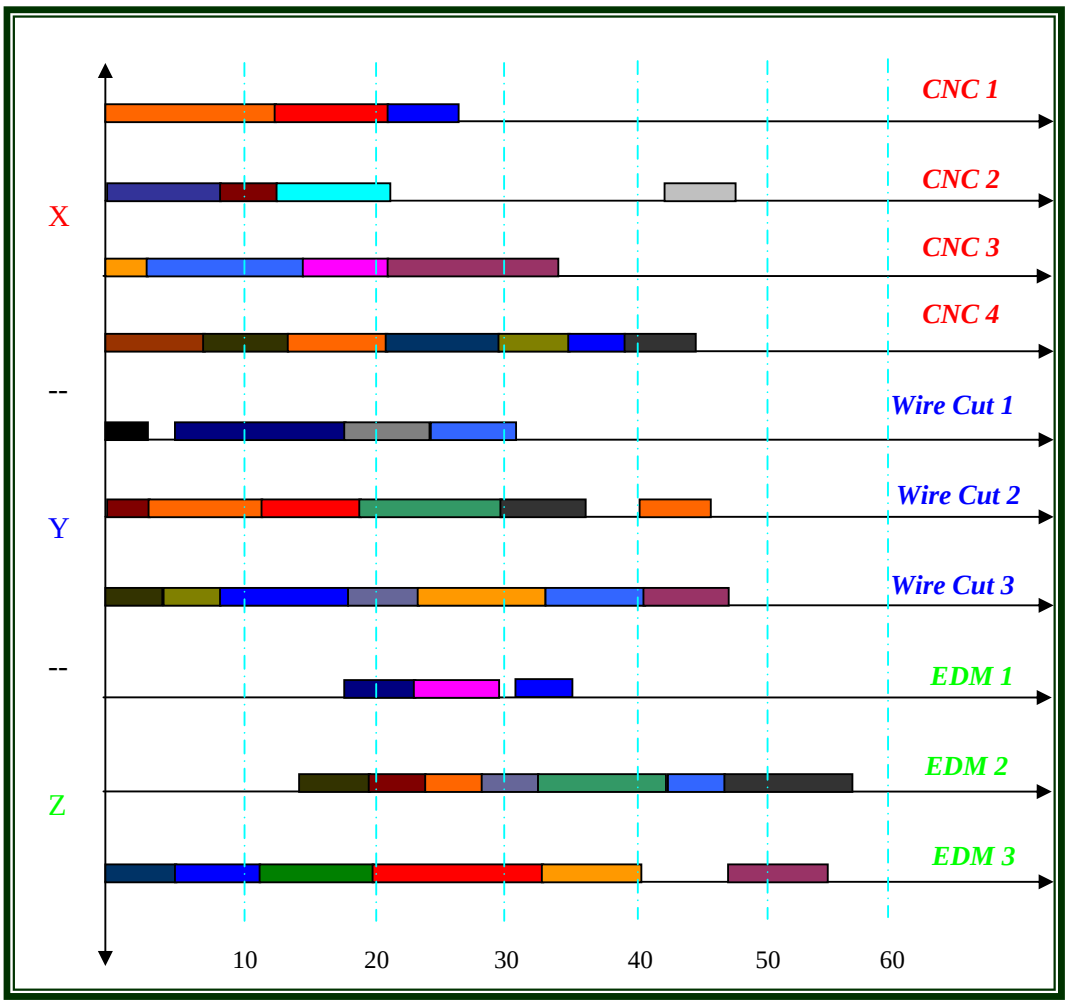

Figure 6.6 Gantt chart for schedule obtained by SA based system for case study II

Details about the jobs used in case study II and some of the results achieved while simulating the system are given in appendix E.

The overall performance of the GA based system for minimizing priority weighted total lateness, total flow time, and combined objectives is presented below.

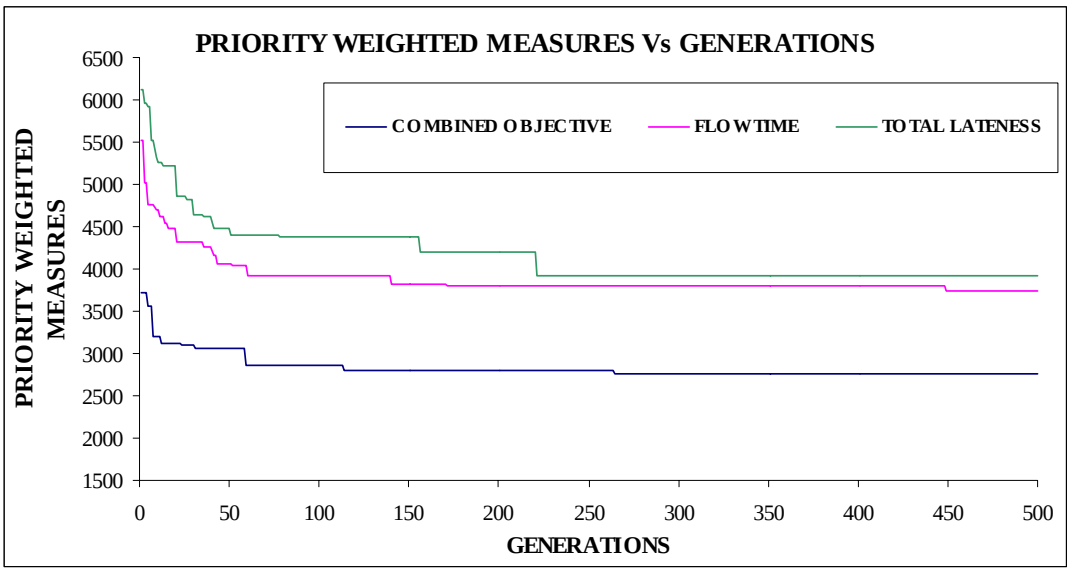

Figure 6.7 Performance of GA based system for priority weighted measures

In the multiple objective performance measure, makespan, flow time and total lateness are combined with three different weights. The sum of all the values is taken as an objective function to be minimized. The performance of the system by simulated annealing algorithm for priority weighted measures is graphically presented as,

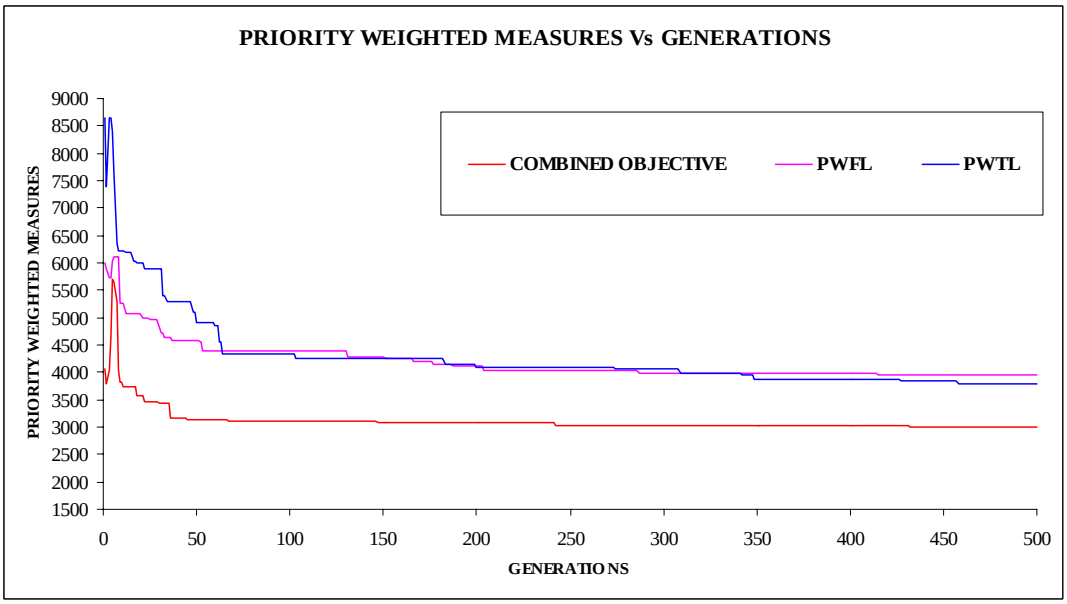

Figure 6.8 Performance of SA based system for priority weighted measures

Some of the Best Schedules of process plans by GA and SA based systems

By the consideration of priority weighted total Flow time as objective function in SA

Execution Time: 0.030 Seconds; Priority weighted total flow time: 3878.0000

Schedule for achieved priority weighted total flow time

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

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

8 5 1 7 8 1 7 9 5 9 4 7

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

5 8 6

By the Consideration of priority weighted total Lateness in GA based system

Execution Time: 0.070 Seconds; Minimum Priority Weighted Lateness: 4034.00

One of the Best Schedule obtained and Priority Weighted Total Lateness

6 4 4 7 10 4 3 10 2 6 10 2

5 10 3 5 10 3 5 3 7 10 9 6

9 5 2 7 8 2 7 9 5 9 4 7

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

6 8 5 *Priority weighted Total Lateness: 4034.00*

By the Consideration of priority weighted combined objective in GA based system

Execution Time: 0.201 Seconds; Minimum Achieved: 2478.90

One of the best schedules achieved and priority weighted combined objective value

6 3 3 7 10 4 4 8 2 7 10 2

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

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

9 4 6 9 10 1 4 7 10 1 1 1

6 10 5 *Priority Weighted Combined Objective: 2478.90*

6.10 COMPARISON OF SYSTEMS

In the case studies, the prototype environment of the mould manufacturing shop is considered for study. The system consists of 25 jobs with their operations and 3 clusters of machines are considered. The system environment is represented with n=25, Nc=3, MNc_X = 4, MNc_Y = 3 and MNc_Z = 3. In the prototype system, cluster X contains 4 CNC machines, Cluster Y contains 3 Wire cut machines and Cluster Z contains 3 EDM machines. Differ from case study I in the case study II priority value is tagged to each job to indicate the importance of job. Four performance measures such as Makespan, total lateness, total flow time and combined objective function are evaluated in the systems. Two Meta heuristic optimization techniques are used to optimize the system. The approaches are tested for the constant number of generation cycles. The performance of the optimization techniques are compared graphically for all the measures. There is not much variation in performance of the optimization techniques while evaluating the measures like total lateness and total flow time. In makespan related objectives population based performance measures are performing better than single search based system. Based on the prototype system study, both the optimization techniques are applicable to schedule the process plans in real time mould manufacturing shop. The system environment and the performance of the Meta heuristic technique for industrial data are explained with case studies in the next chapter.

CHAPTER 7

CASE STUDIES AND DISCUSSIONS

7.1 CASE STUDY I

In this case study, an environment similar to the real mould manufacturing shop is considered for study. The considered mould shop consists of five machining clusters (Departments) namely milling, CNC, Grinding, Wire cut and EDM. The testing starts with all the machines are available for processing first incoming jobs to the particular machines. In total, there are 60 jobs considered throughout the testing period and their full details are given in Appendix F. The experiment is conducted to study performances of four different measures such as makespan, total lateness, total flow time and multiple objective functions in the large scale system environment. Clusters in the considered mould shop contains different machine types, such as the Milling cluster consists of 6 machines, CNC Cluster consists of 9 Machines, Grinding Cluster consists of 6 machines, Wire Cut Cluster consists of 5 machines and EDM Cluster consists of 11 machines. In order to achieve the objective of increasing number of on-time and early jobs in the final schedule, only tardy jobs are penalized with penalty factor 2.0 in the following cases.

Table 7.1 Details of clusters considered in Case Studies

Cluster No	Cluster Machines	Number of Machines
1	Milling	6
2	CNC	9
3	Grinding	6
4	Wire Cut	5
5	EDM	11

In mould manufacturing industries, controlling the tardy jobs are given higher importance than early jobs. By penalizing the tardy jobs in the system, the number of early and on-time jobs will increase in the final schedule. The need of increasing the number of early and on- time job is considered very important in this study. The performance measures which are considered while simulating the systems are,

$$FL_i = (C_i - r_i) \tag{7.1}$$

$$TL_i = \alpha\, EL_i + \beta\, TR_i \tag{7.2}$$

$$EL_i = \max\ \{ d_i - C_i , 0 \} \tag{7.3}$$

$$TR_i = \max\ \{C_i - d_i, 0 \} \tag{7.4}$$

$$\text{Comobject} = \text{Weight}_1 \times \text{Makespan} + \text{Weight}_2 \times \sum_{i=1}^{n} TL_i + \text{Weight}_3 \times \sum_{i=1}^{n} FL_i \tag{7.5}$$

The following parameters and Values are considered in the implemented optimization techniques while studying the system performances:

1) Genetic Algorithm based system

Population size = 10

Recombination: Roulette wheel approach

Crossover probability = 0.85, Mutation Probability = 0.85

Crossover: two point crossover, Mutation: Random selection and picking

Selection Scheme: Elitism with insertion

No of generations: 1000 and 2000

2) Simulated Annealing based system

Initial temperature = 5000

Final temperature = 0.010

Termination condition (generations): 1000 and 2000

Perturbing method: Random selection and picking

The performance of the genetic algorithm based system for flow time and total lateness measures in 1000 iterations are presented graphically as:

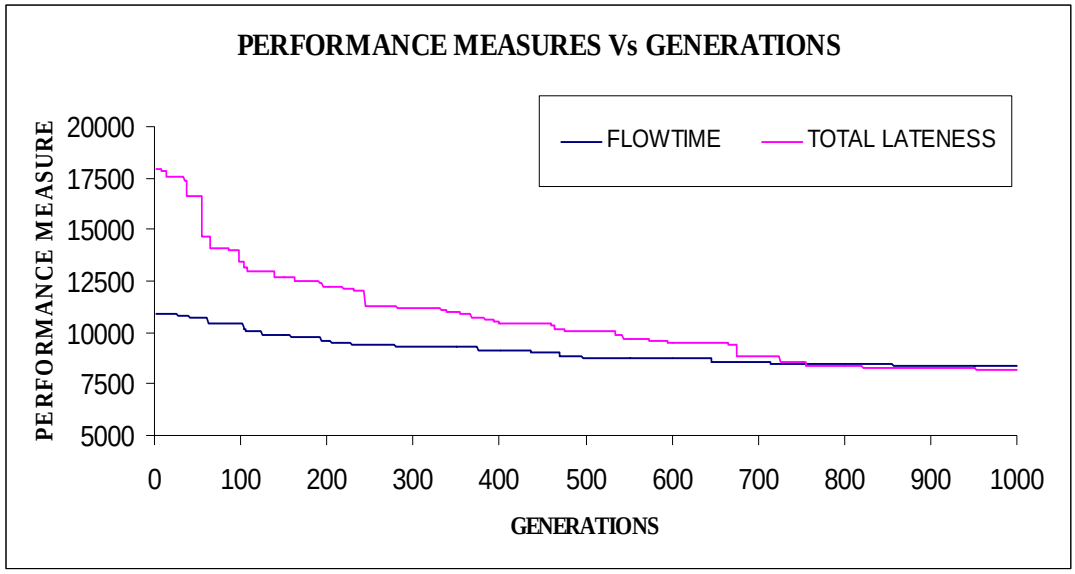

Figure 7.1 Performances of GA based system for Scheduling in mould Shop

The performance of the SA based system for the same total flow time and total lateness measures are presented graphically as:

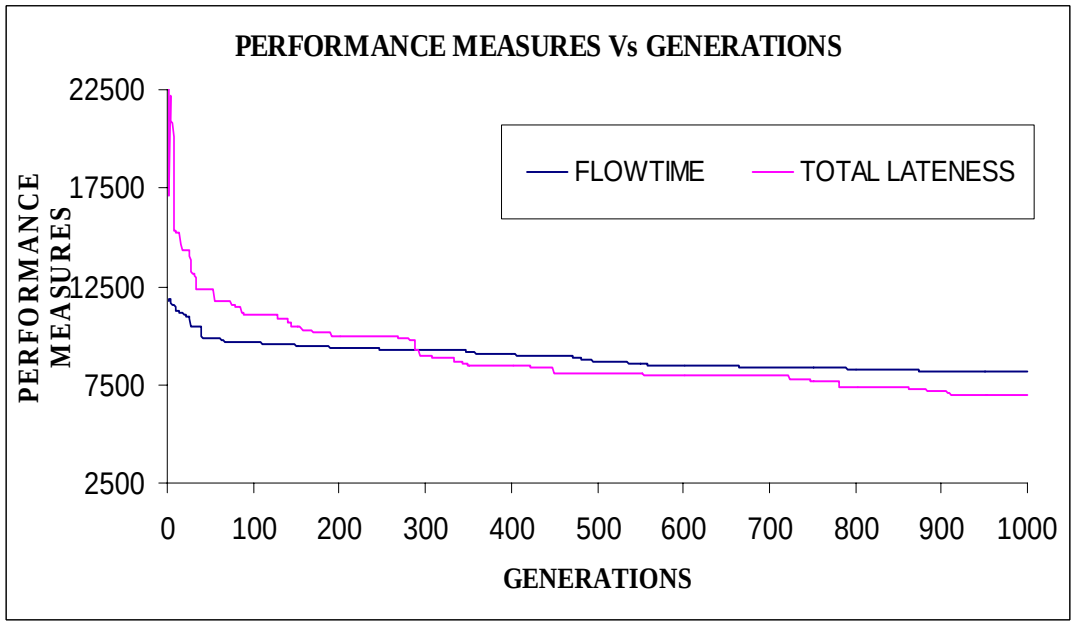

Figure 7.2 Performances of SA based system for Scheduling in mould shop

7.1.1 Some of the best schedules while evaluating Case Study I

The makespan of the overall system by the consideration of 37 machines and 60 jobs in the mould manufacturing shop is studied using Genetic Algorithm and simulated annealing systems. One of the best schedules from 1000^{th} generation cycles and achievable makespan in genetic algorithm based system is given below.

System execution time: 0.340 Seconds

Achieved makespan in 1000 iterations: 238.00 time units

One of the best schedules from 1000^{th} generation cycle and its makespan

8	11	18	11	25	34	5	21	15	35	1	18
35	24	37	8	15	21	11	23	33	1	21	15
24	34	7	13	17	10	23	29	4	19	10	34
26	6	21	26	4	18	22	7	13	35	5	19
7	10	30	8	11	20	14	24	32	7	10	17
11	22	29	3	10	21	15	25	27	6	17	11
36	5	18	26	14	2	15	20	24	12	37	2
21	15	34	4	16	22	3	20	22	35	1	19
26	37	3	17	26	5	21	8	4	21	26	37
7	8	16	8	23	27	7	13	19	12	22	32
1	21	23	32	4	16	26	34	1	18	11	22
5	18	15	35	23	5	19	23	2	7	15	19
25	13	30	3	8	12	18	23	14	33	3	20
11	6	19	8	30	2	17	22	7	31	1	21
24	11	34	3	20	25	15	33	2	16	13	4
19	10	1	15	19	14	36	25	11	27	15	31
17	15	32	20	35	29	31	30	33	35	35	37

25	22	13	22	9	4	12	20	11	24	35	15
9	26	12	30	2	9	13	36	Makespan: 238.00			

The performances of the two optimization techniques for the objective of minimizing Makespan for 1000 iterations are graphically presented as:

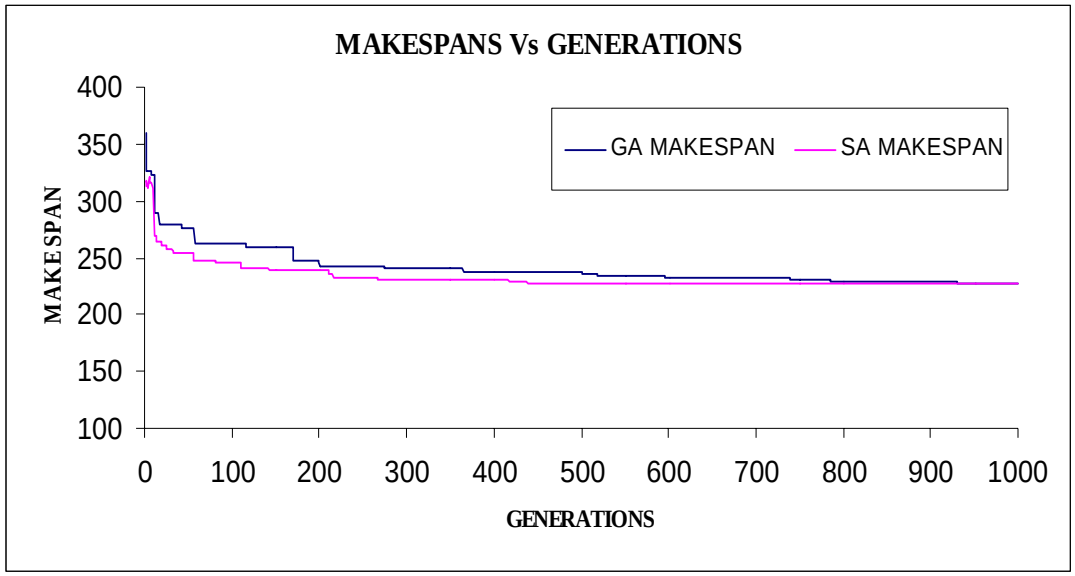

Figure 7.3 Performance of GA and SA based systems for Makespan minimization

The performance of the methods for multiple objective functions for 1000 iterations:

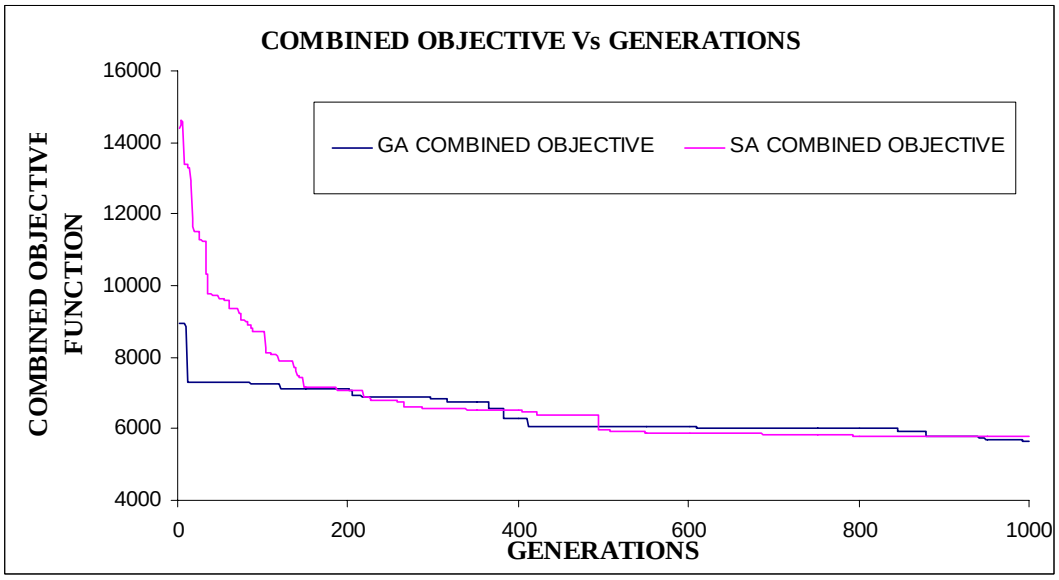

Figure 7.4 Performance of GA and SA based systems for multiple objectives

In the combined objective function, the weights are allocated based on the importance of parameters. The weights of 0.40 for makespan, 0.35 for flow time and 0.35 for total lateness is assigned in the combined objective measure.

7.2 PERFORMANCE EVALUATION OF THE APPROACHES

In this study, scheduling of flexible process plans for the real time mould manufacturing industry is evaluated. Both the simulated annealing algorithm and genetic algorithm is used to perform the scheduling function in mould manufacturing shops. The four performance measures which are considered in this study are makespan, total flow time, total lateness and combined objective function. The variation in performance measures while simulating the scheduling of process plans in mould manufacturing shop is given in Table 7.2.

Table 7.2 Details of variation in measures for 1000 iterations in Case Study I

Performance Measures	Genetic Algorithm	Simulated Annealing
1. Minimum Makespan	227	228
2. Maximum Makespan	359	321
3. Minimum Total Flow time	8339	8196
4. Maximum Total Flow time	10893	11808
5. Minimum Total Lateness	8172	6972
6. Maximum Total Lateness	17937	22659
7. Minimum Combined Objective	5648.6	5773.1
8. Maximum Combined Objective	8940.6	14621.9

From Table 7.2 it is clear that genetic algorithm based system is performing better when searching the makespan and combined objective measures. Meanwhile, simulated annealing algorithm is working well with total flow time and total lateness based

measures. However, these results are obtained from limited search of 1000 generation cycles. Even though the assigned generation cycles are enough for this study, increasing the number of generation cycles may provide better results. The impact of variation in generation cycles is studied in next case study.

7.3 CASE STUDY II

In this study the same environment which is considered in case study I is studied with additional data of priority addition to each job. Priorities are tagged to every job to indicate the importance of finishing that job on time. Priority is assigned to each job from the scale of 1 – 10 based on the necessity of the job. Priority value 1 is assigned to less important job, whereas priority value 10 is tagged to most important job. In terms of performances the makespan based measures do not have the different impact on the priority added jobs in the system. Priority weighted measures are considered as the due date based measure. So the due date based measures such as priority weighted total lateness, and combined objective measures are given importance while simulating the system. In the combined objective function measure, priority weighted total lateness and priority weighted total flow time is considered with the weight of 0.35 along with makespan is assigned with the weight of 0.40. The performance measures used in this case study is presented below:

$$WFL_i = (Ci - r_i) \times P_i \quad (7.6)$$

$$WTL_i = \alpha\, WEL_i + \beta\, WTR_i \quad (7.7)$$

$$WEL_i = \max \{ d_i - C_i , 0 \} \times P_i \quad (7.8)$$

$$WTR_i = \max \{ C_i - d_i , 0 \} \times P_i \quad (7.9)$$

$$\text{Comobject} = \text{Weight}_1 \times \text{Makespan} + \text{Weight}_2 \times \sum_{i=1}^{n} WTL_i + \text{Weight}_3 \times \sum_{i=1}^{n} WFL_i \quad (7.10)$$

WFL_i = Priority Weighted Flow time of Job i,

WTL_i = Priority Weighted Tardiness of Job i,

WEL_i = Priority Weighted Earliness of Job i,

α = Penalty for Early Job;

β = Penalty for Tardy Job

Based on the importance of the criteria, the lateness measures are penalized differently. Similar to the previous case study, the importance of reducing tardiness is given much higher importance than the earliness. Due dates for each operation is calculated from the deadline of each job as the on time delivery of a fully finished job is more critical than achieving high earliness in each operation. The penalty value of 3 is tagged to each tardy job. There is a total of 60 jobs and 5 clusters of machines considered throughout the testing period and their details are given in Appendix F.

The performance of the simulated annealing algorithm for this case is presented graphically as:

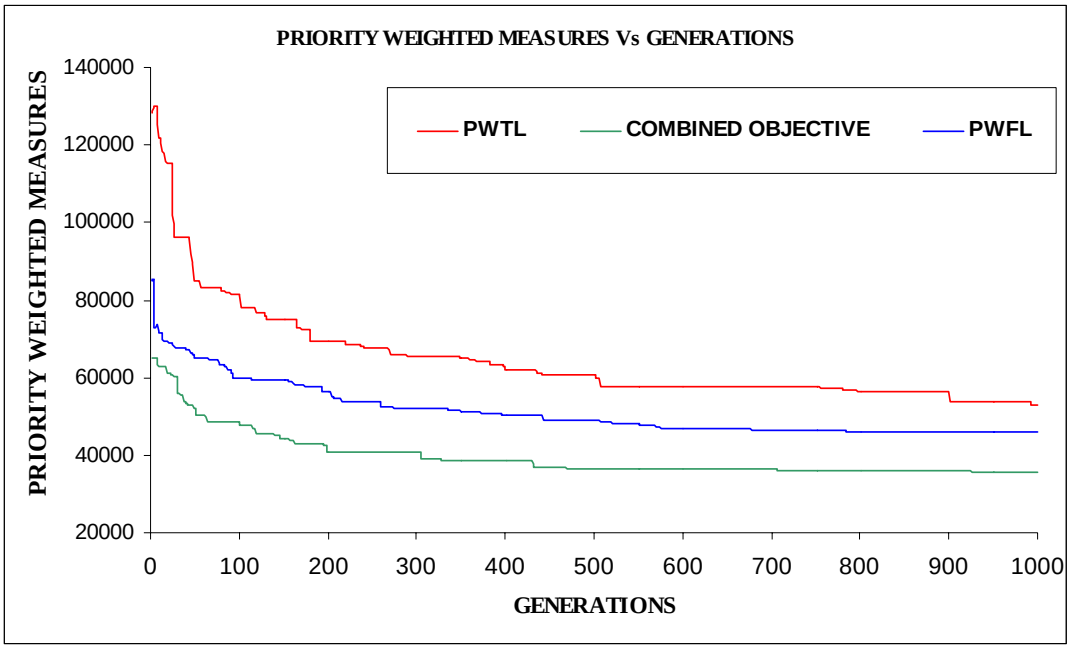

Figure 7.5 Performance of SA based system for priority weighted measures

PWTL represents priority weighted total lateness and PWFL represents the priority weighted total flow time respectively

The Figure 7.5 shows, that the system is not fully converged even in 1000th iteration, so there is chance of obtaining better solutions by increasing the number of generation cycles. In order to evaluate this concept, the system is simulated for 2000 generation cycles. The performance of simulated annealing algorithm based approach for 2000 generation cycles are given in Figure 7.6.

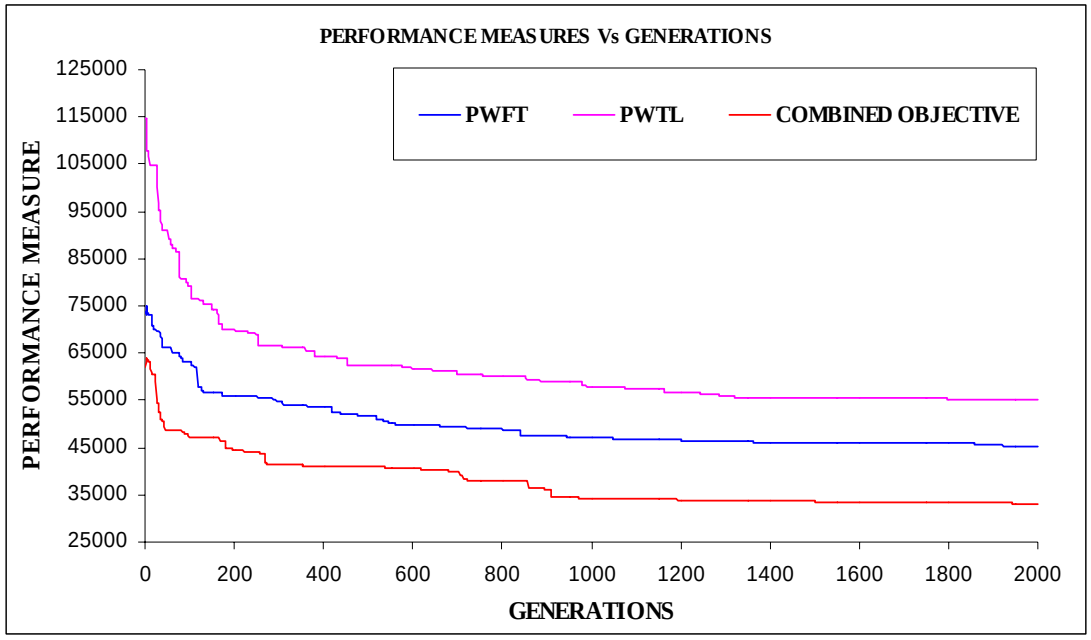

Figure 7.6 Performance of SA based system for 2000 generation cycles

One of the good schedule of process plans while minimizing the priority weighted total flow time measure is given below:

System Execution Time: 0.100 Seconds; PWFL: 46081.0000

Schedule for achieved performance measure

8	14	18	10	25	35	2	20	12	34	2	20
34	24	36	7	15	21	14	23	36	4	21	10
26	34	7	14	17	14	22	31	5	16	12	35
25	2	19	24	5	18	25	8	12	35	5	19
8	13	30	8	14	20	15	26	31	7	11	16
13	22	31	2	12	19	10	23	27	5	18	12

36	2	19	23	14	6	11	21	25	14	36	1
19	11	35	4	17	24	1	17	24	35	4	19
25	36	1	19	22	6	18	7	3	17	24	34
7	7	16	7	22	27	8	10	19	15	22	31
2	17	26	36	5	17	25	30	3	18	12	23
1	18	14	35	23	5	17	24	3	8	14	19
26	12	29	1	8	14	17	25	15	35	4	16
12	2	17	8	30	2	18	22	7	27	1	17
22	13	35	6	18	24	14	34	2	20	12	4
21	10	5	15	19	12	35	22	15	28	14	29
17	14	29	21	36	32	32	32	33	33	33	37
24	23	11	23	9	4	11	16	13	22	35	11
9	23	11	33	6	9	14	35				

Figure 7.7 show the performance of the Genetic Algorithm based system while minimizing the above mentioned performance measures.

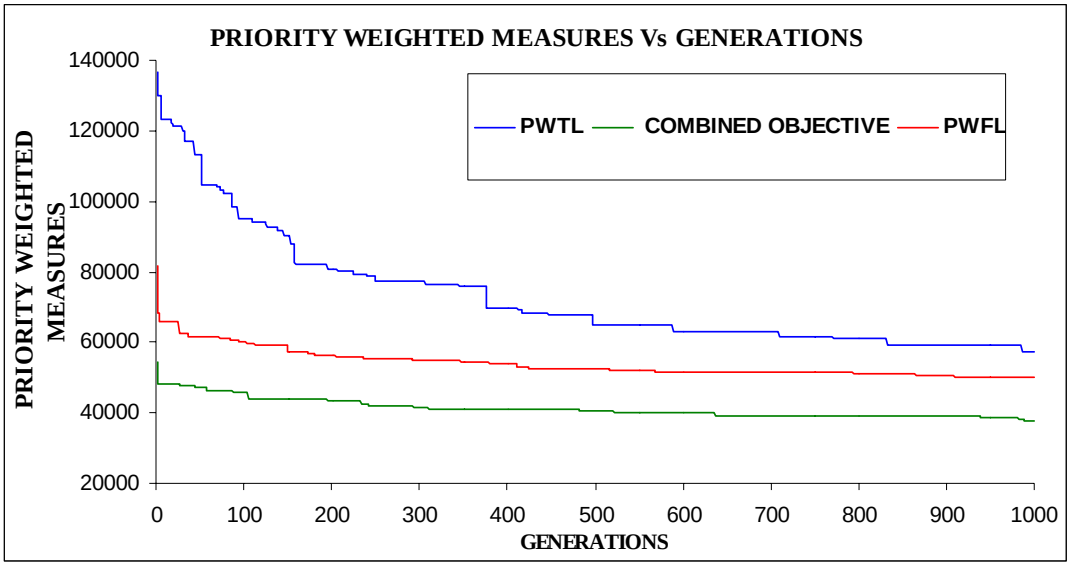

Figure 7.7 Performance of the GA based system for priority weighted measures

Even though the performance of the genetic algorithm based system is better for all the considered measures, from Figure 7.7 shows some disturbances nearer to 1000th generation cycles. To study this condition, genetic algorithm approach is simulated for 2000 iterations and its performance is represented graphically below.

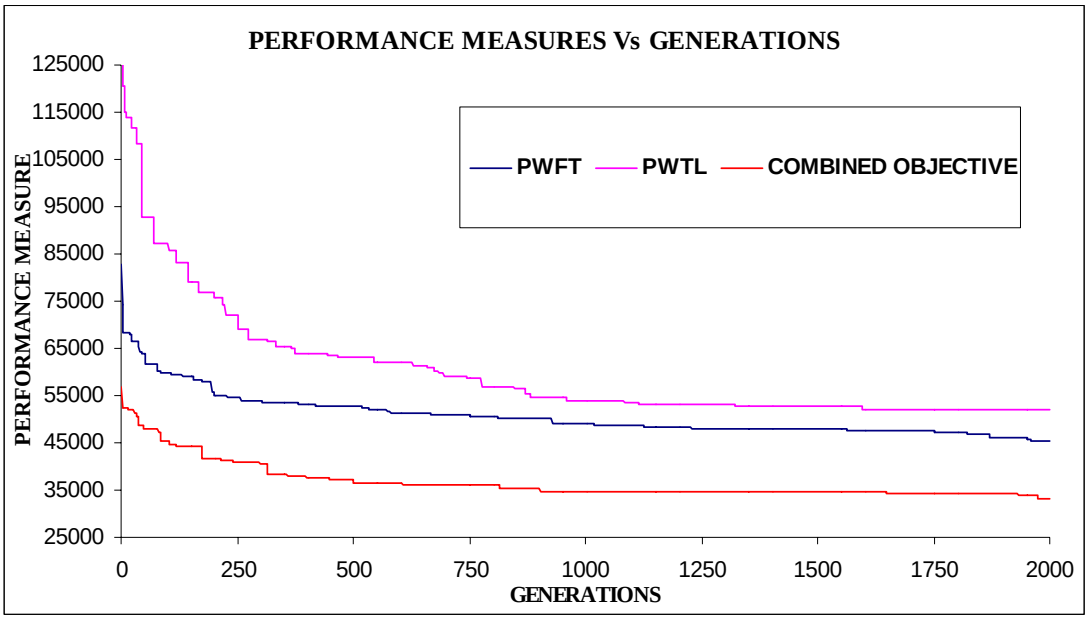

Figure 7.8 Performance of GA based system for 2000 Iterations

Some of the schedules obtained while testing the approaches are given in Appendix F.

7.4 VARIATION IN PERFORMANCE MEASURES

In Case study II, four performance measures are tested by two different Meta heuristic techniques in a mould manufacturing shop. Being differ from case studies performed in last chapter, priority values are tagged to each job to represent the importance of job. In mould manufacturing shops, much importance are given to finishing the jobs in-time or early. So the early jobs are not penalized while searching the better solutions. Variations in performance measures while evaluating the systems for two different iterations are given in Tables 7.3 and 7.4. From simulation results, it seems that population based techniques are performing better with higher number of generation cycles. Crossover and mutation operators plays important role in population based techniques. From studies, it

is proved that selection of appropriate crossover and mutation operators are very important in scheduling studies. Modeling the population based optimization technique with poor perturbing operators converge the solutions very quickly. To avoid this situation it is very important to select the suitable parameters while dealing with scheduling problems.

Table 7.3 Details of variation in measures for 1000 iterations in case study II

Performance Measures	Genetic Algorithm	Simulated Annealing
1. Minimum Makespan	218	229
2. Maximum Makespan	356	340
3. Minimum PWFL	49974	45919
4. Maximum PWFL	81462	85426
5. Minimum PWTL	57428	53067
6. Maximum PWTL	136,882	130078
7. Minimum PW Combined Objective	37773.2	35672.8
8. Maximum PW Combined Objective	54330.8	65211.4

Performances of the makespan based measures are pretty well with genetic algorithm based system than simulated annealing based system. Other three measures are improved quite well with simulated annealing algorithm. From Figures 7.5 and 7.7, it is clear that there is some chance of improvement in the system by increasing the generation cycles. It is proved that increasing the number of generation cycles from 1000 to 2000 increases the performance of the system considerably. The variation in performances of the system for 2000 generation cycles are given in Table 7.4.

Table 7.4 Details of variation in measures for 2000 iterations in Case Study II

Performance Measures	Genetic Algorithm	Simulated Annealing
1. Minimum Makespan	210	229
2. Maximum Makespan	348	340
3. Minimum PWFL	45425	46279
4. Maximum PWFL	82621	75085
5. Minimum PWTL	52049	55133
6. Maximum PWTL	130158	114803
7. Minimum PW Combined Objective	33110	33218
8. Maximum PW Combined Objective	56739.6	63807.1

From the comparison of systems, it is clear that population based Meta heuristic algorithms are performing better than the single solution based search techniques in scheduling flexible process plans in mould manufacturing shop. Compared with the manual factory scheduling and conventional optimization techniques, both the Meta heuristic techniques are superior and efficient in finding the better solutions. However, selecting the appropriate parameters is very important while using the Meta heuristic techniques in planning problems.

CHAPTER 8

CONCLUSIONS AND RECOMMENDATIONS

8.1 CONCLUSIONS

This research focuses on the planning tasks of parallel machine shops and scheduling of flexible process plans in mould manufacturing industries. Sequencing and scheduling considerations prevalent in multiple identical processors with constraints have been addressed in this work. Heuristic techniques are necessary and sometimes only hope to study the critical parameters in single stage or overall structure of the complex systems. In this research, two heuristic algorithms are developed to study the critical parameters in sequencing and scheduling problems of parallel machine systems. One of developed heuristic proved to provide the polynomial time solution for scheduling the identical parallel machine cluster. Finding the optimum solution using conventional optimization techniques will take huge amount of time. Even with long computational time there is no guarantee for the optimum solution. In this research, four Meta heuristic techniques are modified for the suitability of parallel machine environment and simulated for various measures to achieve the better solution. New approach is developed to combine the sequencing and scheduling tasks in parallel machine clusters. The combined approach is tested with the use of memetic algorithm and simulated annealing algorithm in parallel machine clusters.

Another major contribution of this project is the development of suitable scheduling approach which can be used in the flexible mould manufacturing shop. Two Meta heuristic optimization techniques namely simulated annealing and genetic algorithms are applied to solve the scheduling problem in mould manufacturing shop.

The developed approaches are tested in prototype mould shop system and with industrial data. The approaches are tested under different conditions, such as various generation cycles of optimization algorithm and various performance measures considered in the system. The case studies established that the population based algorithm with precisely modeled parameters consistently obtains superior schedules than the present and single solution optimization approaches for various performance measures. Quantitative comparisons between performance measures conducted to better evaluate manufacturing processes. These approaches are very useful to operations managers in order to predict the bottleneck situation in manufacturing environment.

8.2 RECOMMENDATIONS

The sequencing and scheduling of parallel machine cluster is new research topic in planning of manufacturing systems. Machine scheduling researches in manufacturing industries are mainly focused on pure job shop, flow shop and open shop, so chance of developing mathematical model which can demonstrate parallel machine cluster is higher. With the development of pure mathematical model for parallel machine cluster and detailed study of the system, there are some possibilities of predicting the drawbacks in the heuristic algorithms.

In the developed system for scheduling process plans in mould manufacturing shop, there are still possibilities of improving the user interface in the system. To use this system in real time industries, the proposed system should be interfaced with some enterprise solution software. Interfacing the system with enterprise solution software would be the better choice than creating a stand alone system.

There are some possibility of creating the fully integrated process planning and scheduling system by using the proposed approach. Care must be taken while determining the due date for jobs and operations in the system. The time period of mould making is keep on reducing, so developing the time based system to be the better choice for integrated system.

REFERENCES

Adams, J. Balas, E. and Zawack, D. “The shifting bottleneck procedure for job shop scheduling”, Management Science, v34, pp. 391-401,1988.

Alidaee B, “Maximizing set function formulation of two scheduling problems”, ZOR (Zeitschrift fuer Operations Research) Methods and Models of Operations Research, 36, pp 409-416, 1992.

Aldakhilallah K.A and Ramesh R, “Computer-integrated process planning and scheduling (CIPPS): Intelligent support for product design, process planning and control”, International Journal of Production Research, 37(3), pp. 481-500, 1999.

Alting L. and H.C.Zhang, “Computer aided process planning: the start-of-the-art survey”, International Journal of Production Research, 27(4), pp.553-585, 1989.

Avital L and Mosheiov G, “A note on the maximum number of on-time jobs on parallel identical machines”, Computers and Operations Research, 30, pp 1745-1749, 2003.

Baker KR. “Introduction to sequencing and scheduling”, Wiley, New York 1974.

Balakrishnan N, Kanet J J, and Sridharan S V, “Early/tardy scheduling with sequence dependent setups on uniform parallel machines”, Computers and Operations Research, 26, pp 1025-1041, 1999.

Balas E, Lenstra J K, and Vazacopoulos A, “ The one machine problem with delayed precedence constraints and its use in job shop scheduling”, Management Science, 41(1), pp. 94-109, 1995.

Balas E. and Vazacopoulos A, “Guided local search with shifting bottleneck for job shop scheduling”, Management Science, 44(2), pp. 262-253, 1998.

Baptiste P, Jouglet A, Le pape C and Nuijten W, “A constraint-based approach to minimize the weighted number of late jobs on parallel machines”, Technical Report 228, UMR,CNRS 659, France:Heudiasye, 2000.

Barnes J W. and Laguna M, “Solving the multiple machine weighted flow problem using tabu search”, IIE transactions, 25 (2), pp. 121-128, 1993.

Blackstone H G, Phillips E T, and Hogg G L, “The state-of-art survey of dispatching rules for manufacturing job shop operations”, International Journal of Production Research, 20, pp 27-45, 1982.

Biskup D. and T.C. Cheng, “Multiple machine scheduling with earliness, tardiness and completion time penalties,” Computers and Operations Research, v26, pp.45-57, 1999.

Brown, K. and Cagan, J., “Optimized process planning by generative simulated annealing”, Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 11, pp. 219-235, 1997.

Chen S.J. and Li L, “Reducing total tardiness cost in manufacturing cell scheduling by a multi-factor priority rule”, International Journal of Production Research, 37(13), pp. 2939-2956, 1999.

Cheng T C E, and C. C. S. Sin, “A state-of-the art review of parallel machine scheduling research”, European Journal of Operational Research, 77, pp. 271-292, 1990.

Cho, K K. Oh, J S. Ryu, K R. and Choi, “Integrated process planning and scheduling system for block assembly in shipbuilding”, CIRP Anneals, 47(1), pp. 419-422, 1998.

Dogramari A, and Surkis J, “Evaluation of a heuristic for scheduling independent jobs on parallel identical processors”, Management Science, 25, pp. 1208-1216, 1979.

Du J, and J.Y.T.Leung, “Minimizing total tardiness on one machine is NP-hard”, Mathematics of Operations Research, 15, pp.483-495, 1990.

Duflou, J.Kruth, J-P and Van Oudheusden, “Algorithms for the design verification and automatic process planning for bent sheet metal parts”, CIRP Annals, 48(1), pp. 405-408, 1999.

Elmaraghy H.A, “Evalution and future perspectives of CAPP”, Annals of the CIRP, 42(2), pp. 739-751, 1993.

Elmaraghy H.A. and W.H. Elmaraghy, “Bridging the gap between process planning and production planning and control”, In Proc. of the 24th CIRP International seminar on manufacturing systems, Copenhagen, Denmark, pp. 1-10, June 1992.

Faruk C, and C.Constantin, “An IT view on perspectives of computer aided process planning research”, Computers in Industry, 34, pp. 307-337, 1997.

Franca P M, gendreau M, laporte G, and F.M. Muller, “A composite heuristic for the identical parallel machine scheduling problem with minimum makespan objective”, Computers and Operations Research, 21, pp. 205-210, 1994.

Gan P.Y, Scheduling of flexible-sequenced process plans in a mould manufacturing shop, National University of Singapore, 2001.

Gan P.Y, and K.S. Lee, “scheduling of flexible-sequenced process plans in a mould manufacturing shop”, International Journal of Advanced Manufacturing Technology, 20, pp. 214-222, 2002.

Giffler B, and Thompson G, “Algorithms for solving production scheduling problems”, Operations Research VIII, pp. 487-503, July 1960.

Glass C.A, Potts C.N, and Shade P, “Unrelated parallel machine scheduling using local search”, Mathematics and Computer Modeling, 20, pp. 41-52, 1994.

Glover F, “Tabu Search-part I”, ORSA Journal on Computing, 1, pp. 190-206, 1989.

Glover F, “Tabu Search-part II”, ORSA Journal on Computing, 2, pp. 14-32, 1990.

Ho J.C, and Chang Y. L, “Minimizing the number of tardy jobs for m-parallel machines”, European Journal of Operational Research, 26, pp. 127-141, 1999.

Hutchinson G.K, and Pflughoeft K.A, “Flexible Process plans: their value in flexible automation systems”, International Journal of Production Research, 32(3), pp. 707-719, 1994.

Hyun S.C and Kyu H.P, “Shop floor scheduling at shipbuilding yards using multiple intelligent agent system”, Journal of Intelligent manufacturing, v8, pp. 505-515, 1997.

Jawahar N, P.Aravindan, S.G. Ponnambalam, and A. Aravind Karthikeyan, “A genetic algorithm based scheduler for set-up constrained FMC”, Computers in Industry, v35, pp. 291-310, 1998.

Kim K-H, and Egbelu P.J, “Scheduling in production environment with multiple process plans per job”, International Journal of Production Research, 37(12), pp. 2725-2753, 1999.

Kirkpatrick S, Gelatt C.D.Jr, and Vecchi M P, “Optimization by Simulated Annealing”, Science, 220, pp. 671-680, 1995.

Koulamas C.P, “The total tardiness problem: review and extensions”, Operations Research Letters, 42, pp. 1025-1041, 1994.

Lai, K. K. and Chan, W. M., “Developing a simulated annealing algorithm for the cutting stock problem”, Computers and Industrial Engineering, 32(1), pp.115-127, 1997.

Lee H.F, “Production planning for flexible manufacturing systems with multiple machine types: a practical approach”, International Journal of Production Research, 36(10), pp. 2911-2927, 1998.

Lenstra J.K, A.H.G. Rinnooy Kan, and P.Brucker, “Complexity of machine scheduling problems”, Annals of Discrete Mathematics, 1, pp. 343-362, 1977

Li C.L, “A heuristic for parallel machine scheduling with agreeable due dates to minimize the number of late jobs”, Computers and Operations Research, 22, pp. 277-283, 1995

Li C and Cheng T, “The parallel machine min-max weighted absolute lateness scheduling problem”, Naval Research Logistics, 41, pp. 33-46, 1993.

Lundy M, and A.Mees, “Convergence of an annealing algorithm”, Math. Prog, 34, pp. 111-124, 1986.

Ma, G. H., Zhang, Y. F., and Nee, A. Y. C., “A simulated annealing-based optimisation algorithm for process planning”, International Journal of Production Research, 38 (12), pp. 2671-2687, 2000.

Metropolis W, A. Rosenbluth , M. Rosenbluth, M.Teller, and A. Teller, “Equations of state calculations by fast computing machines”, Journal of Chem.Physics,21, pp. 1087-1092, 1953..

Montgomery C. Douglas, “Design and analysis of experiments”, New York, Wiley, 1984

Moscato P, “On evolution search, optimization, genetic algorithms and martial arts: Towards memetic algorithms, Technical Report, C3P, 826, Caltech concurrent computation program, 1989.

Moore J.M, “An n Job, one machine sequencing algorithm for minimizing the number of tardy jobs”, Management Science, 15, pp. 102-109, 1968.

Naumann A, and Gu P, “Real-time part dispatching within manufacturing cells using fuzzy logic”, Production planning and Control, 8(7), pp. 662-669, 1997

Piewitt W, and D.Biskup, “A note on an efficient algorithm for the single machine tardiness problem”, International Journal of Production Economics, 66, pp. 287-292, 2000.

Saravanakumar M, and Lee K.S., “Comparing meta heuristic algorithms in scheduling on mould manufacturing shop”, In Proc. of the 3rd International Conference on Advanced Manufacturing Technology, Kuala Lumpur, Malaysia, pp. 473-481, May 11-13, 2004.

Saravanakumar M, and Lee K. S., “Intelligent optimization techniques in sequencing and scheduling of parallel machines”, accepted for publication in International Conference on Manufacturing Automation, Wuhan, China, October 26-29, 2004.

Szu H, “Fast simulated annealing, neural networks for computing: American Institute of Physics, pp. 420-425, In. J. Denver, (Ed.) New York, 1986.

Wang H.P, and J.K.Li, “Intelligent reasoning for Process planning”, Computers in Industry, 8, pp. 293- 309, 1987.

Zhang H.C and L.Alting, “Computerized Manufacturing Process planning Systems”, 1st Edition, Chapman & Hall, 1994.

Zhou C, and P.Egbelu, “Scheduling in a manufacturing shop with sequence dependent setups”, Robotics and Computer Integrated Manufacturing, 5(1), pp. 73-81, 1989.

APPENDIX A

In this Appendix, the explanation for heuristic algorithm II (HC_{II}) is presented. The two phase heuristic algorithm is explained with 2 machines and 10 jobs environment.

Explanation for HC_{II}:

In this problem, the parallel machine shop consists of 2 CNC machines and 10 jobs is considered with the objective function of maximizing the number of in-time and early jobs. It is assumed that each job is ready for processing at the beginning of the scheduling horizon with distinct processing time and distinct due date. The setup and handling time of each job are included with the processing time.

Table A.1 Details of jobs with slack time

Jobs	1	2	3	4	5	6	7	8	9	10
t_i	11	18	6	12	21	10	14	28	13	19
d_i	14	40	46	48	46	22	16	38	30	23
s_i	3	22	40	36	25	12	2	10	17	4

S_i is the starting time for job i that assures the completion of job without lateness

It is generally known as slack time available for job i, ($S_i = d_i - t_i$).

Phase I

STEP 1:

1) Arrange jobs in ascending order of S_i

Table A.1 (a). Details of jobs arranged by ascending order of slack time

Jobs	1	2	3	4	5	6	7	8	9	10
t_i	14	11	19	28	10	13	18	21	12	6
d_i	16	14	23	38	22	30	40	46	48	46
s_i	2	3	4	10	12	17	22	25	36	40

2) Jobs are numbered based on current order

3) $\{k_j\} = \{14, 11\}$; $n = 10$ (Number of Jobs)

$a = 2$; $\{G_s\} = \Phi$; $m = 2$ (Number of machines)

STEP 2:

Table A.1 (b). Procedure to perform Step 2 in Phase I of HC_{II}

$\{L_s\}_0$	a	(K_j , S_a)	(a , *n*)	{ k_j }	$\{L_s\}_f$
{1,2}	3	11>4	$3 \neq 10$	{14,11}	{1,2}
{1,2}	4	11>10	$4 \neq 10$	{14,11}	{1,2}
{1,2}	5	11<12	$5 \neq 10$	{14,21}	{1,2,5}
{1,2,5}	6	14<17	$6 \neq 10$	{27,21}	{1,2,5,6}
{1,2,5,6}	7	21<22	$7 \neq 10$	{27,39}	{1,2,5,6,7}
{1,2,5,6,7}	8	27>25	$8 \neq 10$	{27,39}	{1,2,5,6,7}
{1,2,5,6,7}	9	27<36	$9 \neq 10$	{39,39}	{1,2,3,6,7,9}
{1,2,5,6,7,9}	10	39<40	10 =10	{45,39}	{1,2,3,6,7,9,10}

Using this step the main procedure is repeated for n-m times to get the initial local schedule. With this procedure the schedule can be generated in polynomial time. This polynomial time heuristic produces the most possible number of in-time or early jobs in this local schedule.

STEP 3: Unscheduled jobs {3, 4, and 8}

Phase II

STEP 1:

$\{G_s\} = \{1, 2, 5, 6, 7, 9, \text{and } 10\}$

$\{L_s\} = \Phi$

Jobs	3	4	8
t_i	19	28	21
d_i	23	38	46
S_i	4	10	25

$\{K_j\}= \{19, 28\}$; $\{L_s\} = \{3, 4\}$; n=3;

a = 2; n > a; Go to Step 2 of Phase I

a = 3; 3=3; 19<25; $\{L_s\} = \{3, 4, 8\}$

$\{G_s\} = \{1,2,5,6,7,9,10\} \cup \{3,4,8\}$

STEP 2:

$\{SS\}_w$ Before Sequencing = {{1, 2}, {7, 9}, {4, 8}}

After sequencing, schedule on ascending order of processing time

{{2, 1}, {7, 9},{8,4}}

$\{SS\}_e$ Before Sequencing = {{5, 6}, {10, 3}}

After sequencing, schedule based on descending order of processing time

{{6, 5}, {3, 10}}

Final schedule {{1, 2}, {6, 5}, {7, 9}, {3, 10}, {4, 8}}

Ordered Jobs for Machine 1: {1, 6, 7, 3, 4}

Ordered Jobs for Machine 2: {2, 5, 9, 10, 8}

APPENDIX B

In this appendix, the results for cases considered in chapter 4 are presented.

Table B.1 Performance of Job-code system for priority weighted total lateness measure

LEVELS / SAMPLES	GA	MA	TS	SA
6*2	512,0.512	512,0.520	512,0.270	512,0.24
10*2	1440,0.620	1524,0.610	1440,0.370	1440,0.12
20*2	8756,1.001	9374,1.161	8876,0.721	8840,0.11
25*2	12170,1.321	13200,1.502	12236,1.011	12318,0.23
Mean value	5719.5,0.866	6152.5,0.948	5766,0.593	5777.5,0.175
10*3	716,0.620	720,0.691	752,0.320	716,0.100
20*3	4394,1.031	4444,0.201	4440,0.731	4282,0.18
30*3	11934,1.652	13006,1.872	18168,1.392	11960,0.13
40*3	27220,2.433	30934,2.803	32998,2.353	27410,0.19
Mean Value	11066,1.434	12276,1.392	14089.5,1.20	11092,0.15
10*5	420,0.620	420,0.65	662,0.370	420,0.20
20*5	2762,1.051	2886,1.191	4398,0.741	2730,0.21
35*5	16458,2.053	20038,2.433	25416,1.762	16288,0.19
45*5	25642,2.934	31172,3.564	35220,2.884	25920,0.16
Mean Value	11320.5,1.66	13629,1.9595	16424,1.439	11339.5,0.19
25*8	1572,1.382	1594,1.642	3872,1.081	1618,0.21
35*8	3814,2.063	3732,2.613	8104,1.872	3704,0.18
45*8	8888,2.994	9644,3.834	18450,3.084	8736,0.21
55*8	11502,4.256	12702,5.376	19386,4.33	11686,0.24
Mean value	6444,2.674	6918,3.366	12453,2.59	6436,0.21
25*10	1006,1.382	1006,1.742	1934,1.091	1008,0.21
35*10	2928,2.063	2960,2.713	7444,1.852	2910,0.33
50*10	7222,3.605	7114,4.735	18748,3.605	6818,0.17
80*10	23454,9.123	26816,11.526	45846,9.133	24148,0.33
Mean value	8652.5,4.043	7474,5.179	18493,3.92	8721,0.26

Each field is assigned with [a, b]; which represents a = Priority weighted total lateness achieved in 1000 iterations and b = computational time required for each algorithm.

Levels = Algorithms taken for study

Sample= problem instances. In problem set 6*2, where, 6 represents number of jobs and 2 represent number of machines.

Table B.2 Performance of Machine-code system for total lateness measure

LEVELS / SAMPLES	GA	MA	TS	SA
6*2	306,0.440	306,0.43	358,0.210	388,0.190
10*2	1016,0.550	1016,0.55	1102,0.38	1038,0.20
20*2	5156,0.771	5372,0.7810	4822,0.701	5910,0.190
25*2	8258,0.891	8258,0.9210	7040,0.871	8294,0.130
Mean value†	3684,0.663	3738,0.671	3330.5,0.541	3907.5,0.178
10*3	512,0.663	522,0.49	650,0.330	512,0.270
20*3	3246,0.711	3248,0.781	3230,0.691	3206,0.110
30*3	8464,0.951	8398,1.011	8486,1.211	10742,0.440
40*3	17912,1.171	17924,1.331	16866,1.942	18096,0.280
Mean Value	7533.5,0.600	7523,0.903	7308,1.044	8139,0.275
10*5	352,0.530	364,0.59	342,0.44	660,0.140
20*5	2436,0.821	2400,0.741	2362,0.681	2520,0.11
35*5	9498,1.091	9684,1.161	9132,1.472	10768,0.20
45*5	15762,1.23	16112,1.462	14710,2.293	16126,0.18
Mean Value	7012,1.452	7140,0.989	6636.5,1.222	7518.5,0.158
25*8	1748,0.861	1588,0.901	2074,0.981	2050,0.16
35*8	2764,1.031	2792,1.181	3460,1.552	3296,0.17
45*8	5406,1.251	5632,1.411	6402,2.343	6544,0.18
55*8	7012,1.452	6616,1.682	7690,3.274	6970,0.220
Mean value	4232.5,1.149	4157,1.294	4906.5,2.038	4715,0.1825
25*10	666,0.891	666,0.961	1028,0.941	756,0.22
35*10	2078,1.131	2100,1.171	2708,1.602	2694,0.13
50*10	4288,1.452	4244,1.552	5814,2.844	4600,0.14
80*10	14914,2.143	15060,2.403	15652,7.14	16978,0.190
Mean value	5486.5,1.404	5517.5,1.522	6300.5,3.241	6257,0.17

† Mean value in tables indicates the mean value of performances and computational time for similar number of machines.

Table B.3 Performance of Machine-code system for priority weighted total lateness measure

LEVELS / SAMPLES	GA	MA	TS	SA
6*2	750,0.440	750,0.440	862,0.30	750,0.18
10*2	1862,0.50	1988,0.500	2032,0.47	2401,0.14
20*2	12122,0.761	12122,0.791	11368,0.63	12158,0.16
25*2	18842,0.931	18842,0.961	16232,0.881	19952,0.16
Mean value	8394,0.658	8425.5,0.673	7623.5,0.570	8815.5,0.16
10*3	956,0.590	800,0.500	1204,0.38	800,0.250
20*3	5070,0.841	4964,0.791	6164,0.65	6682,0.12
30*3	13712,0.980	13652,1.061	15092,1.161	15368,0.25
40*3	34912,1.181	34912,1.301	36410,2.032	36722,0.30
Mean Value	13662.5,0.898	13582,0.913	14717.5,4.22	14893,0.23
10*5	492,0.540	492,0.50	678,0.35	846,0.15
20*5	3450,0.751	3738,0.801	4340,0.66	3900,0.26
35*5	18696,1.121	20126,1.181	21900,1.562	20544.0.13
45*5	30158,1.341	30048,1.392	29644,2.453	31432,0.26
Mean Value	13199,0.938	13601,0.969	14140.5,1.26	14180.5,0.2
25*8	3322,0.921	3704,0.921	4846,0.891	5424,0.22
35*8	6266,1.081	5610,1.171	8888,1.532	7722,0.13
45*8	10194,1.351	10436,1.432	14016,2.423	12152,0.20
55*8	14020,1.552	12860,1.682	17310,3.505	13826,0.21
Mean value	8450.5,1.226	8152.5,1.302	11265,2.088	9781,0.19
25*10	1130,0.891	1070,0.961	2168,1.011	1546,0.17
35*10	3282,1.071	3188,1.171	4638,1.682	5940,0.20
50*10	9610,1.502	9774,1.622	20598,3.034	14108,0.42
80*10	29210,2.133	29000,6.00	48160,7.931	37140,0.22
Mean value	10808,1.3993	10759.5,1.44	18891,3.415	14683.5,0.26

Table B.4 Performance of Job-code system for total lateness measure

LEVELS / SAMPLES	GA	MA	TS	SA
6*2	294,0.460	294,0.490	326,0.290	294,0.150
10*2	912,0.610	912,0.650	912,0.500	912,0.140
20*2	4536,0.961	4572,1.141	4746,0.680	4542,0.120
25*2	6634,1.291	6692,1.512	6944,1.011	6656,0.380
Mean value	3094,0.831	3117.5,0.948	3232,0.620	3101,0.198
10*3	474,0.590	472,0.681	508,0.620	472,0.170
20*3	2894,1.021	2880,1.171	3162,0.33	2888,0.290
30*3	7948,1.602	7930,1.872	11354,0.731	7962,0.240
40*3	15702,2.353	15774,2.794	18024,1.382	15818,0.220
Mean Value	6754.5,1.392	6764,1.630	8262,1.162	6785,0.23
10*5	318,0.600	318,0.681	404,0.330	318,0.100
20*5	2236,1.051	2172,1.231	2858,0.771	2172,0.170
35*5	8940,1.942	9068,2.403	11168,1.802	8902,0.210
45*5	14210,2.870	14518,3.555	18124,2.784	14138,0.390
Mean Value	6426,1.62	6519,1.968	8139,1.422	6382.5,0.218
25*8	1084,1.301	998,1.968	1430,1.051	992,0.180
35*8	2214,2.053	2196,1.672	3432,1.802	2186,0.150
45*8	5126,2.914	5048,2.623	6890,2.834	5078,0.300
55*8	6412,4.176	6346,3.805	10098,4.156	6368,0.300
Mean value	3709,2.611	3647,3.357	5462.5,2.461	3656,0.2325
25*10	602,1.341	592,1.732	786,1.041	598,0.140
35*10	1862,1.992	1852,2.684	3852,1.832	1876,0.260
50*10	3820,3.55	3678,4.707	6474,3.465	3680,0.280
80*10	13874,9.123	13540,11.577	20512,8.562	13672,0.440
Mean value	5039.5,4.001	4915.5,5.175	7906,3.725	4956.5,0.280

APPENDIX C

In this appendix, the randomly generated data for the cases considered in chapter 5 are presented. t_i = Processing time for job i, P_i = Priority value for job i

Table C.1 Randomly generated data set for 100 jobs and 14 machines case

Job No	t_i	P_i	Job No	t_i	P_i	Job No	t_i	P_i
1	2	3	41	6	2	81	9	3
2	9	1	42	13	3	82	10	1
3	4	2	43	8	1	83	6	1
4	6	3	44	5	1	84	10	1
5	12	1	45	12	1	85	4	2
6	13	2	46	1	1	86	11	3
7	11	1	47	4	1	87	8	4
8	10	1	48	7	1	88	15	3
9	13	2	49	13	2	89	13	4
10	1	2	50	2	2	90	7	2
11	9	1	51	15	4	91	15	4
12	6	2	52	2	4	92	13	4
13	9	4	53	4	3	93	8	3
14	15	3	54	4	3	94	8	1
15	12	2	55	13	1	95	9	3
16	15	2	56	4	3	96	12	1
17	8	2	57	1	1	97	12	1
18	10	1	58	1	2	98	1	1
19	3	4	59	8	2	99	7	1
20	15	2	60	14	4	100	5	4
21	4	3	61	7	2			
22	6	1	62	1	4			
23	5	4	63	12	1			
24	6	4	64	8	4			
25	13	2	65	12	2			
26	10	1	66	8	4			
27	11	4	67	11	1			
28	5	3	68	11	2			
29	5	1	69	12	4			
30	3	3	70	8	1			
31	2	3	71	10	1			
32	1	2	72	5	1			
33	4	1	73	7	1			
34	1	2	74	11	1			
35	10	2	75	2	2			
36	6	1	76	9	4			
37	4	4	77	14	1			
38	6	2	78	12	3			
39	9	3	79	1	2			
40	1	1	80	13	2			

Table C.2 Randomly generated data set for 150 jobs and 14 machines case

Job no	t_i	P_i	Job no	t_i	P_i	Job no	t_i	P_i
1	12	4	51	11	4	101	10	3
2	9	3	52	3	3	102	3	3
3	8	2	53	11	4	103	6	2
4	11	3	54	8	4	104	8	4
5	13	1	55	6	1	105	4	2
6	12	4	56	8	4	106	3	4
7	9	4	57	1	1	107	2	3
8	2	1	58	10	4	108	13	1
9	2	1	59	12	1	109	12	3
10	4	1	60	8	1	110	9	1
11	11	4	61	5	2	111	3	1
12	14	1	62	6	3	112	5	1
13	3	1	63	13	1	113	13	1
14	7	2	64	4	1	114	1	2
15	15	1	65	15	2	115	12	4
16	1	2	66	6	2	116	4	4
17	9	1	67	14	4	117	3	3
18	1	4	68	2	1	118	11	1
19	8	2	69	14	3	119	7	1
20	1	1	70	10	2	120	2	3
21	2	2	71	6	2	121	3	3
22	1	3	72	10	3	122	8	3
23	1	4	73	4	2	123	14	1
24	4	3	74	4	2	124	14	2
25	1	1	75	1	1	125	1	2
26	10	1	76	10	1	126	1	1
27	1	2	77	7	1	127	3	1
28	10	3	78	12	4	128	8	4
29	11	3	79	1	1	129	7	2
30	3	1	80	3	1	130	1	1
31	13	4	81	1	1	131	9	1
32	3	1	82	13	1	132	6	2
33	11	1	83	7	1	133	4	1
34	8	3	84	10	1	134	4	1
35	6	1	85	1	1	135	8	2
36	5	2	86	6	1	136	1	3
37	15	1	87	14	2	137	10	1
38	7	1	88	8	1	138	1	2
39	7	4	89	7	3	139	12	2
40	9	1	90	9	1	140	4	3
41	8	1	91	1	4	141	15	2
42	4	1	92	10	3	142	14	1
43	9	2	93	12	2	143	7	2
44	11	2	94	1	4	144	2	1
45	14	1	95	15	3	145	14	3
46	6	4	96	1	3	146	6	1
47	15	2	97	8	1	147	11	1
48	12	4	98	3	3	148	1	1
49	10	3	99	15	1	149	4	3
50	7	1	100	2	4	150	7	4

Table C.3 Randomly generated data set for 100 jobs and 8 machines Case

Job i	t_i	P_i	Job i	t_i	P_i	Job i	t_i	P_i
1	1	2	41	6	3	81	13	2
2	14	1	42	11	4	82	13	2
3	3	2	43	6	1	83	8	1
4	1	4	44	9	1	84	8	4
5	4	4	45	12	1	85	6	3
6	15	1	46	4	3	86	2	2
7	8	1	47	1	2	87	15	3
8	14	1	48	7	2	88	1	2
9	2	4	49	4	1	89	8	3
10	9	2	50	6	2	90	2	3
11	7	1	51	15	3	91	8	2
12	6	1	52	3	4	92	14	4
13	1	1	53	11	4	93	9	3
14	10	2	54	10	1	94	13	3
15	1	3	55	12	3	95	1	1
16	15	2	56	14	1	96	8	1
17	7	4	57	3	1	97	1	4
18	5	1	58	8	3	98	3	1
19	6	1	59	1	3	99	14	4
20	7	3	60	13	1	100	10	3
21	4	4	61	1	4			
22	15	1	62	15	4			
23	1	2	63	14	2			
24	12	1	64	7	3			
25	9	2	65	11	3			
26	3	4	66	1	3			
27	10	2	67	6	2			
28	4	2	68	8	4			
29	10	4	69	2	2			
30	15	3	70	4	3			
31	3	4	71	15	2			
32	7	3	72	1	1			
33	1	1	73	6	2			
34	5	3	74	3	3			
35	13	4	75	5	3			
36	1	1	76	6	1			
37	9	1	77	12	2			
38	11	3	78	10	1			
39	15	4	79	7	1			
40	15	1	80	15	1			

Table C.4 Randomly generated data set for 150 jobs and 8 machines case

Job No	t_i	P_i	Job No	t_i	P_i	Job No	t_i	P_i
1	5	1	51	14	3	101	2	3
2	15	3	52	9	2	102	1	4
3	1	3	53	1	1	103	15	1
4	9	4	54	8	1	104	2	3
5	6	4	55	2	2	105	1	4
6	4	1	56	15	2	106	15	1
7	12	1	57	13	4	107	12	1
8	9	1	58	4	4	108	2	4
9	1	1	59	10	1	109	9	2
10	14	1	60	2	1	110	13	3
11	1	4	61	2	2	111	15	4
12	2	3	62	1	2	112	1	1
13	5	2	63	15	2	113	15	1
14	11	4	64	14	3	114	12	2
15	9	4	65	3	1	115	4	3
16	13	4	66	1	3	116	8	3
17	3	1	67	12	4	117	9	3
18	13	4	68	14	2	118	3	3
19	1	4	69	6	1	119	12	4
20	7	2	70	10	4	120	11	1
21	1	1	71	2	3	121	1	2
22	13	1	72	3	1	122	14	1
23	7	2	73	6	2	123	9	4
24	1	1	74	10	1	124	15	2
25	10	3	75	4	2	125	4	1
26	5	1	76	4	1	126	11	2
27	1	1	77	1	3	127	5	1
28	1	3	78	1	2	128	12	1
29	12	1	79	12	1	129	6	2
30	5	1	80	9	2	130	13	2
31	5	3	81	6	4	131	12	2
32	1	3	82	13	1	132	1	1
33	10	2	83	4	4	133	11	1
34	1	1	84	15	3	134	13	3
35	2	1	85	15	2	135	13	1
36	10	1	86	10	2	136	11	4
37	8	2	87	14	1	137	15	4
38	12	1	88	7	2	138	4	1
39	6	2	89	4	1	139	13	3
40	1	1	90	1	4	140	4	3
41	1	4	91	13	1	141	15	1
42	5	1	92	7	2	142	2	1
43	6	3	93	5	1	143	4	1
44	9	4	94	2	1	144	2	2
45	4	2	95	9	3	145	14	3
46	10	3	96	7	1	146	10	1
47	10	2	97	2	1	147	14	1
48	13	1	98	15	4	148	4	4
49	2	1	99	15	1	149	1	1
50	14	2	100	5	1	150	4	1

APPENDIX D

Total number of Jobs: 25
Nc = 3, MNc_X = 4, MNc_Y = 3 and MNc_Z = 3.

Table D.1 Details of jobs in prototype mould shop without priority

Job (J_i)	d_i	Op	Cluster X			Cluster Y				Cluster Z			Prec Job	Prec Op
			1	2	3	4	5	6	7	8	9	10		
1	17	1					3	3						
2	14	1	7	7	7	7								
3	32	1	6	6	6	6								
		2							4					
		3								5	5	5		1,2
4	26	1	7	7	7	7								
5	38	1		8	8	8								
		2								5	5	5		
6	47	1	8	8										
		2					12	12	12					1
		3								5	5	5		2
7	33	1	4	4	4	4								
		2					3	3	3					
		3								4	4	4		1,2
8	36	1	12	12	12	12								
		2					8	8						
		3									4	4		1,2
9	34	1	5	5	5	5								
		2					4	4	4					
10	43	1	4	4	4	4								
		2					9	9	9					
		3									6	6		
11	21	1								8	8	8		
12	35	1					5	5	5					
		2								4	4	4		1
13	23	1					6	6						
14	41	1	8	8										
		2					7	7	7					
		3								12	12	12		1,2
15	47	1	3	3	3	3								
		2					9	9	9					
		3								7	7	7		1,2
16	42	1					10	10	10					
		2								9	9	9		1
17	45	1			8	8								
		2							7					
		3									4	4		1,2
18	41	1	5	5	5	5								
		2					6	6						
		3									9	9		1,2
19	21	1								6	6	6		
20	15	1	6	6	6	6								
21	37	1	12	12	12	12								
		2					4	4	4					
		3									7	7	20	1,2
22	12	1	8	8										
23	15	1	5	5	5	5							15	
24	30	1	5	5	5	5								
		2					6	6						
		3								4	4	4		1,2
25	14	1					5	5					10	

SAMPLE SOLUTION FOR GA BASED SYSTEM WITH OBJECTIVE OF MINIMIZING MAKESPAN OF THE ENTIRE SYSTEM

********* GA BASED SYSTEM FOR SCHEDULING FLEXIBLE PROCESS PLANS *******

********* MAKESPAN AS PERFORMANCE MEASURE *******

********* PARAMETERS AND PROBLEM CONSIDERED *******

POPULATION SIZE: 10
CROSSOVER PROBABILITY: 0.85
MUTATION PROBABILITY : 0.80
EARLY JOB PENALTY : 2.00
TARDY JOB PENALTY : 3.00
TOTAL NUMBER OF JOBS: 25
TOTAL NUMBER OF MACHINES IN SYSTEM: 10
TOTAL NUMBER OF GENERATIONS: 1000
OPTIMIZING PERFORMANCE MEASURE OF MAKESPAN IN THIS SYSTEM BY GA

TWO OF THE BEST SCHEDULES IN INITIAL POPULATION AND FITNESS VALUES

5 4 2 7 10 3 3 8 2 5 9 4 5
10 4 6 9 1 7 4 5 9 10 5 8 6
2 7 9 4 6 10 5 10 4 7 9 3 6
9 8 3 4 5 10 2 3 4 6 8 5
74.00
6 2 4 7 10 2 3 10 1 6 8 3 6
9 4 6 9 1 7 2 7 9 8 7 10 5
2 7 8 4 7 10 7 8 3 7 9 2 5
10 8 4 4 6 9 2 3 4 6 10 6
74.00

GENERATION NUMBER: 1

Two of the schedules in Population after Recombine

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

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

Four of the schedules in Population after crossover

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

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

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

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

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

Two of the Schedules in Population after mutation

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

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

Two of the schedules in Population for next generation and fitness

5 4 2 7 10 3 3 8 2 5 9 4 5
10 4 6 9 1 7 4 5 9 10 5 8 6
2 7 9 4 6 10 5 10 4 7 9 3 6
9 8 3 4 5 10 2 3 4 6 8 5
74.00
6 2 4 7 10 2 3 10 1 6 8 3 6
9 4 6 9 1 7 2 7 9 8 7 10 5
2 7 8 4 7 10 7 8 3 7 9 2 5
10 8 4 4 6 9 2 3 4 6 10 6
74.00

GENERATION NUMBER: 1000

Two of the better schedules in Population after Recombine

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

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

Two of the schedules in Population after crossover

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

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

Four of the schedules in Population after mutation

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

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

Two of the schedules in Population for next generation and fitness

6 1 4 7 10 1 4 8 2 6 10 3 7
9 4 6 10 2 7 2 5 10 8 7 9 6
2 7 9 1 5 8 5 8 3 7 9 2 5
10 8 1 1 7 9 2 3 4 6 8 6
48.00

6 1 4 7 10 1 4 9 2 6 10 3 7
9 4 6 10 2 7 2 5 10 8 7 9 6
2 7 9 1 5 8 5 8 3 7 9 2 5
10 8 1 4 6 9 2 3 4 6 8 6
49.00

EXECUTION TIME: 1.255 Seconds

GLOBAL MAKESPAN: 48.00

SOME OF THE BEST SCHEDULES AND MAKESPANS

6 1 4 7 10 1 4 8 2 6 10 3 7
9 4 6 10 2 7 2 5 10 8 7 9 6
2 7 9 1 5 8 5 8 3 7 9 2 5
10 8 1 1 7 9 2 3 4 6 8 6
48.00
6 1 4 7 10 1 4 9 2 6 10 3 7
9 4 6 10 2 7 2 5 10 8 7 9 6
2 7 9 1 5 8 5 8 3 7 9 2 5
10 8 1 4 6 9 2 3 4 6 8 6
49.00
6 1 4 7 10 1 4 9 2 6 10 3 7
9 4 6 10 2 7 2 5 10 8 7 9 6
2 7 9 1 5 8 5 8 3 7 9 2 5
10 8 1 4 6 9 2 3 4 6 8 6
49.00

APPENDIX E

Total Number of Jobs=25, Nc=3, MNc_x = 4, MNc_Y = 3, and MNc_Z = 3.

Table E.1 Details of jobs in prototype mould shop with priority

Job	p_i	d_i	Op	Cluster X				Cluster Y			Cluster Z			Prec	Prec
				1	2	3	4	5	6	7	8	9	10	Job	Op
1	5	17	1					3	3						
2	5	14	1	7	7	7	7								
3	6	32	1	6	6	6	6								
			2							4					
			3								5	5	5		1,2
4	6	26	1	7	7	7	7								
5	5	38	1		8	8	8								
			2								5	5	5		
6	5	47	1	8	8										
			2					12	12	12					1
			3								5	5	5		2
7	5	33	1	4	4	4	4								
			2					3	3	3					
			3								4	4	4		1,2
8	6	36	1	12	12	12	12								
			2					8	8						
			3									4	4		1,2
9	5	34	1	5	5	5	5								
			2					4	4	4					
10	4	43	1	4	4	4	4								
			2					9	9	9					
			3									6	6		
11	4	21	1								8	8	8		
12	9	35	1					5	5	5					
			2								4	4	4		1
13	5	23	1					6	6						
14	7	41	1	8	8										
			2					7	7	7					
			3								12	12	12		1,2
15	6	47	1	3	3	3	3								
			2					9	9	8					
			3								7	7	7		1,2
16	5	42	1					10	10	10					
			2								9	9	9		1
17	5	45	1			8	8								
			2							7					
			3									4	4		1,2
18	5	41	1	5	5	5	5								
			2					6	6						
			3									9	9		1,2
19	5	21	1								6	6	6		
20	6	15	1	6	6	6	6								
21	6	37	1	12	12	12	12								
			2					4	4	4					
			3									7	7	20	1,2
22	9	12	1	8	8										
23	10	15	1	5	5	5	5							15	
24	6	30	1	5	5	5	5								
			2					6	6						
			3								4	4	4		1,2
25	7	14	1					5	5					10	

SAMPLE RESULTS OF THE PERFORMANCE OF SA BASED SYSTEM WITH THE OBJECTIVE OF MINIZING PRIORITY WEIGHTED TOTAL LATENESS

****SA SYSTEM FOR BASED SCHEDULING FLEXIBLE PROCESS PLANS * ***
******** TOTAL LATENESS AS PERFORMANCE MEASURE ******
******** PARAMETERS AND PROBLEM CONSIDERED ******

INITIAL TEMPERATURE: 5000.00
FINAL TEMPERATURE : 0.0100
TARDY JOB PENALTY : 3.00
EARLY JOB PENALTY : 2.00
TOTAL NUMBER OF JOBS: 25
TOTAL NUMBER OF MACHINES IN SYSTEM: 10
OPTIMIZATION OF PRIORITISED TOTAL LATENESS BY SA

Generation Number: 1
Temperature: 5000.000000
Current schedule

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

Perturbing Jobs: 21 5 19
Modified Schedule

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

Cost of Initial schedule: 9154
Cost of Modified schedule: 9998

REJECTED

Generation Number: 2
Temperature: 5000.000000
Current schedule

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

Perturbing Jobs: 6 19 9

Modified Schedule

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

Cost of Initial schedule: 9154
Cost of modified schedule: 9555
Probability: 0.089 Acceptance Probabilities: 0.923

POOR BUT ACCEPTED

Cost of Initial schedule: 3548
Cost of modified schedule: 4794

REJECTED

Generation Number: 1000
Temperature: 0.010012
Current schedule

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

Perturbing Jobs: 24 4 17
Modified Schedule

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

Cost of Initial schedule: 3548
Cost of modified schedule: 4574

REJECTED

EXECUTION TIME 0.120 Seconds
GLOBAL MAKESPAN: 3548
SCHEDULE FOR GLOBAL MAKESPAN

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

APPENDIX F

STRUCTURE OF THE SYSTEM

Table F.1 Details of machine clusters in mould shop

Cluster	Number of Machines
Milling	6
CNC	9
Grinding	6
Wire Cut	5
EDM	11

Total number of Machines in entire system: 37

Total number of Jobs considered: 60

Penalty for early job: 0.0

Penalty for tardy job: 2.0

Priority scale for jobs: 1-10

1- Lowest weight and 10- Highest weight

Considered parameters and values in GA and SA

	GENETIC ALGORITHM	SIMULATED ANNEALING
Population Size	10	1
Generations	500, 1000	500, 1000
Perturbation	Two point crossover (85%) Random change mutation (80%)	Random select and pick

Nc=5, MNc_1 = 6, MNc_2 = 9, MNc_3 = 6, MNc_4 = 5 and MNc_5 = 11.

Total number of Jobs considered in the system: 60
Total number of Clusters considered : 5

Table F.2 Details of jobs taken from mould manufacturing shop (Gan P Y, 2001)

Job	Priority	Deadline	Op ID	Operation	Machining times	Job Prec	Op Prec
1	5	168	1	CNC-Drilling	20 20 0 0 0 0 0 0 0		
			2	CNC-Roughing	0 0 0 6 6 6 6 6 6		1
			3	Grinding	0 10 10 10 0 0		2
			4	CNC-Finishing	0 0 0 10 10 10 10 10 10		3
			5	WireCut	25 25 25 25 25		4
			6	EDM	0 0 0 0 0 0 30 30 30 0 0		5
2	5	168	1	Milling	10 10 10 10 10 10		
			2	Grinding	0 0 0 0 10 10		1
			3	CNC	0 0 0 10 10 10 10 10 10		2
			4	EDM	0 0 0 0 0 0 6 6 6 0 0		3
3	5	168	1	Milling	10 10 10 10 10 10		
			2	Grinding	10 10 10 10 10 10		1
			3	EDM	0 0 0 0 0 0 6 6 6 0 0		2
4	5	168	1	WireCut	0 0 20 20 20		
			2	EDM	0 0 0 0 0 0 0 0 0 16 16		1
5	5	168	1	CNC-Drilling	10 10 0 0 0 0 0 0 0		
			2	CNC-Roughing	0 0 0 6 6 6 6 6 6		1
			3	Grinding	0 0 0 0 8 8		2
			4	CNC-Finishing	0 0 0 5 5 5 5 5 5		3
			5	WireCut	15 15 15 15 15		4
			6	EDM	0 0 0 0 0 0 6 6 6 6 6		4
6	5	168	1	Milling	4 4 4 4 4 4	1	
			2	Grinding	0 0 0 0 5 5		1
			3	CNC	0 0 0 2 2 2 2 2 2		2
			4	WireCut	3 3 3 3 3		3
			5	EDM	0 0 0 0 0 0 1 1 1 1 1		3
7	6	168	1	CNC-Drilling	20 20 0 0 0 0 0 0 0		
			2	CNC-Roughing	0 0 0 10 10 10 10 10 10		1
			3	Grinding	0 10 10 10 0 0		2
			4	CNC-Finishing	0 0 0 10 10 10 10 10 10		3
			5	WireCut	10 10 0 0 0		4
			6	EDM	0 0 10 10 10 10 0 0 0 0 0		4
8	6	168	1	Milling	5 5 5 5 5 5		
			2	Grinding	10 10 10 10 10 10		1
			3	CNC	0 0 0 6 6 6 6 6 6		2
			4	EDM	0 0 0 0 0 0 10 10 10 0 0		3
			5	WireCut	4 4 4 4 4		3
9	6	168	1	Milling	5 5 5 5 5 5		
			2	Grinding	5 5 5 5 5 5		
			3	WireCut	5 5 5 5 5		
10	6	168	1	Milling	3 3 3 3 3 3		
			2	Grinding	0 7 7 7 0 0		1
			3	WireCut	8 8 8 8 8		2
			4	CNC-Drilling	3 3 0 0 0 0 0 0 0		2
			5	CNC-R&F	0 0 0 4 4 4 4 4 4		2
			6	EDM	0 0 0 0 0 0 3 3 3 0 0		5

Table F.2 Contd.,

Job	Priority	Deadline	Op ID	Operation	Machining times	Job Prec	Op Prec
11	6	168	1	Milling	3 3 3 3 3 3		
			2	Grinding	0 7 7 7 0 0		1
			3	CNC-Drilling	3 3 0 0 0 0 0 0 0		2
			4	CNC-R&F	0 0 0 4 4 4 4 4 4		2
			5	EDM	0 0 4 4 4 4 4 4 4 0 0		4
12	6	168	1	CNC-Drill	20 20 0 0 0 0 0 0 0	2,3	
			2	CNC-Rough	0 0 0 10 10 10 10 10 10		1
			3	Grinding	10 10 10 10 0 0		2
			4	CNC-Finish	0 0 0 7 7 7 7 7 7		3
			5	WireCut	17 17 17 17 17		4
			6	EDM	0 0 20 20 20 20 0 0 0 0 0		4
13	6	168	1	CNC-Drilling	10 10 0 0 0 0 0 0 0		
			2	CNC-Roughing	0 0 0 10 10 10 10 10 10		1
			3	Grinding	15 15 15 15 0 0		2
			4	CNC-Finishing	0 0 0 8 8 8 8 8 8		3
			5	WireCut	20 20 0 0 0		4
			6	EDM	0 0 20 20 20 20 0 0 0 0 0		5
14	6	168	1	Milling	5 5 5 5 5 5	4	
			2	CNC	0 0 0 3 3 3 3 3 3		1
			3	Grinding	7 7 7 7 7 7		2
			4	CNC	0 0 0 3 3 3 3 3 3		3
			5	WireCut	3 3 3 3 3		4
			6	EDM	3 3 3 3 3 3 3 3 3 3 3		4
15	6	168	1	Milling	3 3 3 3 3 3		
			2	Grinding	4 4 4 4 4 4		1
			3	CNC	0 0 0 4 4 4 4 4 4		2
			4	EDM	0 0 0 0 0 0 0 0 3 3		3
16	6	168	1	Milling	3 3 3 3 3 3	6	
			2	Grinding	3 3 3 3 3 3		1
			3	WireCut	3 3 3 3 3		2
			4	CNC	0 0 0 3 3 3 3 3 3		3
17	6	168	1	Milling	3 3 3 3 3 3		
			2	CNC	0 0 0 3 3 3 3 3 3		1
			3	Grinding	0 3 3 3 3 3		2
			4	WireCut	0 0 3 3 3		3
			5	CNC	0 0 0 3 3 3 3 3 3		3
			6	EDM	0 0 0 0 0 0 0 0 3 3		5
18	6	168	1	Milling	3 3 3 3 3 3		
			2	Grinding	3 3 3 3 3 3		1
			3	CNC	0 0 0 4 4 4 4 4 4		2
			4	EDM	0 0 0 0 0 0 4 4 4 4 4		3
19	6	168	1	Milling	3 3 3 3 3 3		
			2	Grinding	3 3 3 3 3 3		1
			3	WireCut	3 3 3 3 3		2
20	6	168	1	Milling	3 3 3 3 3 3	10	
			2	Grinding	5 5 5 5 5 5		1
			3	WireCut	4 4 4 4 4		2
			4	EDM	0 0 0 0 0 0 4 4 4 4 4		3

Table F.2 Contd.,

Job	Priority	Deadline	Op ID	Operation	Machining times	Job Prec	Op Prec
21	6	168	1	Milling	8 8 8 8 8 8		
			2	Grinding	6 6 6 6 6 6		1
			3	WireCut	8 8 8 8 8		2
			4	EDM	0 0 0 0 0 0 3 3 3 3 3		3
22	6	168	1	Milling	10 10 10 10 10 10	11	
			2	Grinding	10 10 10 10 10 10		1
			3	WireCut	10 10 10 10 10		2
23	6	168	1	Milling	4 4 4 4 4 4		
			2	Grinding	4 4 4 4 4 4		1
			3	CNC	4 4 4 4 4 4 4 4 4		2
24	6	168	1	Milling	7 7 7 7 7 7		
			2	Grinding	8 8 8 8 8 8		1
			3	WireCut	10 10 10 10 10		2
			4	EDM	0 0 0 0 0 0 3 3 3 3 3		3
25	5	168	1	CNC-Drilling	20 20 0 0 0 0 0 0 0	6,7	
			2	CNC-Roughing	30 30 0 0 0 0 0 0 0		1
			3	Grinding	10 0 0 0 0 0		2
			4	CNC-Finishing	10 10 0 0 0 0 0 0 0		3
			5	WireCut	5 5 0 0 0		4
			6	EDM	10 10 0 0 0 0 0 0 0 0 0		4
26	5	168	1	CNC-Drilling	20 20 0 0 0 0 0 0 0		
			2	CNC-Roughing	0 0 0 20 20 20 20 20 20		1
			3	Grinding	0 10 10 10 0 0		2
			4	CNC-Finishing	0 0 0 15 15 15 15 15 15		3
			5	WireCut	10 10 0 0 0		4
			6	EDM	0 0 15 15 15 15 0 0 0 0 0		4
27	5	168	1	Milling	5 5 5 5 5 5		
			2	Grinding	5 5 5 5 5 5		1
			3	WireCut	3 3 3 3 3		2
			4	EDM	0 0 3 3 3 3 3 3 3 3 3		2
28	5	168	1	Milling	5 5 5 5 5 5		
			2	Grinding	5 5 5 5 5 5		1
			3	WireCut	3 3 3 3 3		2
			4	EDM	0 0 3 3 3 3 3 3 3 3 3		2
29	5	168	1	Milling	5 5 5 5 5 5	10	
			2	Grinding	5 5 5 5 5 5		1
			3	CNC	0 0 0 3 3 3 3 3 3		2
			4	WireCut	3 3 3 3 3		2
30	5	168	1	Milling	5 5 5 5 5 5		
			2	Grinding	0 5 5 5 0 0		1
			3	CNC	0 0 0 3 3 3 3 3 3		2
			4	EDM	0 0 0 0 0 0 3 3 3 0 0		3
			5	WireCut	3 3 3 3 3		3

Table F.2 Contd.,

Job	Priority	Deadline	Op ID	Operation	Machining times	Job Prec	Op Prec
31	5	168	1	Milling	5 5 5 5 5 5		
			2	Grinding	0 5 5 5 0 0		1
			3	WireCut	3 3 3 3 3		2
32	5	168	1	Milling	8 8 8 8 8 8		
			2	CNC	3 3 0 0 0 0 0 0 0		1
			3	CNC	0 0 0 8 8 8 8 8 8		2
			4	Grinding	0 8 8 8 0 0		3
			5	WireCut	5 5 5 5 5		4
			6	CNC	5 5 5 5 5 5 5 5 5		4
			7	EDM	0 0 5 5 5 5 0 0 0 0 0		6
33	5	168	1	Milling	8 8 8 8 8 8	9	
			2	CNC	3 3 3 0 0 0 0 0 0		1
			3	CNC	0 0 0 8 8 8 8 8 8		2
			4	Grinding	5 5 5 5 5 5		3
			5	WireCut	5 5 5 5 5		4
			6	CNC	0 0 0 5 5 5 5 5 5		4
			7	EDM	0 0 0 0 0 0 5 5 5 0 0		6
34	5	168	1	Milling	3 3 3 3 3 3		
			2	Grinding	4 4 4 4 4 4		1
			3	CNC	0 0 0 4 4 4 4 4 4		2
35	5	168	1	Milling	5 5 5 5 5 5	10	
			2	Grinding	5 5 5 5 0 0		1
			3	CNC	4 4 0 0 0 0 0 0 0		2
			4	EDM	0 0 4 4 4 4 0 0 0 0		3
36	5	168	1	Milling	5 5 5 5 5 5		
			2	Grinding	0 5 5 5 0 0		1
			3	WireCut	5 5 0 0 0		
			4	CNC	4 4 0 0 0 0 0 0 0		2
			5	EDM	4 4 4 4 4 4 0 0 0 0		4
37	5	168	1	Milling	5 5 5 5 5 5	25	
			2	Grinding	3 3 3 3 3 3		1
			3	WireCut	3 3 3 3 3		2
			4	CNC	0 0 0 3 3 3 3 3 3		2
			5	EDM	0 0 0 0 0 0 4 4 4 4 4		4
38	5	168	1	Milling	5 5 5 5 5 5		
			2	Grinding	3 3 3 3 3 3		1
			3	WireCut	3 3 3 3 3		2
			4	CNC	0 0 0 3 3 3 3 3 3		2
			5	EDM	0 0 0 0 0 0 4 4 4 0 0		4
39	5	168	1	Milling	5 5 5 5 5 5		
			2	Grinding	5 5 5 5 5 5		1
			3	CNC	0 0 0 5 5 5 5 5 5		2
40	5	168	1	Milling	5 5 5 5 5 5		
			2	Grinding	5 5 5 5 5 5		1
			3	CNC	0 0 0 5 5 5 5 5 5		2

Table F.2 Contd.,

Job	Priority	Deadline	Op ID	Operation	Machining times	Job Prec	Op Prec
41	5	168	1	Milling	5 5 5 5 5 5		
			2	CNC	0 0 0 5 5 5 5 5 5		1
			3	Grinding	8 8 8 8 8 8		2
			4	CNC	0 0 0 4 4 4 4 4 4		3
			5	EDM	0 0 0 0 0 0 3 3 3 3 3		4
			6	WireCut	4 4 4 4 4		3
42	8	40	1	CNC	0 0 0 4 4 4 4 4 4	4	
			2	EDM	15 15 0 0 0 0 0 0 0 0 0		1
43	8	30	1	CNC	0 0 0 10 10 10 10 10 10		
			2	EDM	0 0 8 8 8 8 0 0 0 0 0		1
44	8	40	1	Grinding	0 6 6 6 0 0		
			2	CNC	0 0 0 5 5 5 5 5 5		1
			3	EDM	0 0 13 13 13 13 0 0 0 0 0		2
45	8	30	1	Grinding	0 0 0 0 6 6		
			2	EDM	0 0 0 0 0 0 13 13 13 13 13		1
46	8	20	1	EDM	0 0 15 15 15 15 0 0 0 0 0		
47	5	20	1	EDM	0 0 6 6 6 6 0 0 0 0 0		
48	5	42	1	EDM	0 0 12 12 12 12 0 0 0 0 0		
49	8	15	1	EDM	0 0 0 0 0 0 7 7 7 0 0		
50	5	30	1	EDM	0 0 0 0 0 0 9 9 9 0 0		
51	10	15	1	EDM	0 0 0 0 0 0 13 13 13 0 0		
52	8	25	1	EDM	0 0 0 0 0 0 0 0 0 15 15		
53	8	10	1	WireCut	0 0 5 5 5		
54	8	15	1	WireCut	3 3 0 0 0		
55	7	10	1	CNC	0 0 0 10 10 10 10 10 10		
			2	WireCut	6 6 0 0 0		1
56	10	10	1	CNC	0 0 6 0 0 0 0 0 0		
57	8	40	1	Milling	5 5 5 5 5 5		
			2	CNC	0 0 0 3 3 3 3 3 3		1
			3	Grinding	5 5 5 5 5 5		2
			4	CNC Finishing	0 0 0 7 7 7 7 7 7		3
			5	WireCut	3 3 3 3 3		4
			6	EDM	0 0 0 0 0 0 13 13 13 0 0		4
58	6	20	1	CNC	0 0 0 7 7 7 7 7 7		
			2	CNC Drilling	0 0 5 0 0 0 0 0 0		
			3	WireCut	4 4 4 4 4		
59	8	10	1	CNC	0 0 0 7 7 7 7 7 7		
			2	EDM	0 0 0 9 9 9 9 9 0 0		1
60	6	40	1	Milling	5 5 5 5 5 5		
			2	CNC Drilling	0 0 8 0 0 0 0 0 0		1
			3	CNC R&F	0 0 0 12 12 12 12 12 12		2
			4	EDM	0 0 0 0 0 0 8 8 8 8 8		3

Performance of the approaches without Priority Addition

In this study, the above mentioned mould shop data is considered without priority. Column two in the Table F.2 represents the priority value assigned to each job.

Performance of SA based system for overall Makespan minimization is given below:

Execution time: 1.141 Seconds; Minimum Makespan achieved: 217 time units.

Schedule for the achieved Makespan

8	13	17	15	26	33	5	21	14	34	6	21
34	24	37	7	11	21	12	26	37	3	21	11
23	34	7	12	17	13	22	31	4	19	13	35
26	3	18	24	6	19	26	8	14	33	1	19
8	15	30	7	10	18	12	25	32	8	13	19
10	23	32	4	11	16	11	25	27	4	16	11
37	3	21	24	10	1	15	19	25	14	37	1
19	14	36	5	20	26	1	17	25	33	3	20
25	35	5	16	23	5	19	12	4	18	25	36
8	8	16	8	22	28	7	11	18	10	22	29
4	20	25	29	6	20	23	30	3	17	14	24
3	19	13	33	23	3	19	22	3	7	15	18
24	11	32	6	7	14	20	26	15	35	6	17
12	3	18	7	29	6	17	23	7	28	2	17
23	11	37	6	20	26	14	33	3	21	11	5
19	11	3	15	18	12	35	24	12	28	10	31
19	13	31	20	34	32	29	30	34	34	35	36
24	22	14	23	9	3	14	19	10	24	33	10
9	23	13	30	5							

The achievable schedule using simulated annealing algorithm for Multiple objective function measure in 1000 generation cycles is given below:

Execution time: 0.081 Seconds

Achieved minimum Combined Objective: 5581.1000

Achievable schedule

8	10	19	10	25	35	6	20	12	33	4	21
33	24	37	8	14	20	14	25	33	1	20	12
26	37	8	10	17	10	22	29	2	16	15	33
23	4	17	24	6	17	22	7	13	35	6	19
7	14	35	7	13	19	12	25	32	8	15	17
11	23	32	6	10	17	10	25	33	5	20	11
36	3	16	22	13	4	14	20	26	11	37	2
21	11	35	4	21	23	6	21	22	36	5	16
24	36	5	18	22	5	17	13	4	21	22	37
7	7	16	7	22	27	8	14	17	13	23	29
4	17	24	33	5	18	24	36	6	21	13	23
4	19	11	33	24	6	17	25	5	8	11	19
24	10	32	1	8	14	17	25	11	33	4	17
13	2	16	7	29	2	19	22	7	27	1	21
24	13	37	4	19	25	10	33	2	19	11	5
19	12	2	12	21	11	37	24	12	28	15	32
17	12	29	20	35	30	31	30	34	34	34	36
26	23	14	23	9	6	14	18	14	26	35	10
9	25	15	30	2	9	15	36				

Performance of the approaches with Priority Addition

One of the best schedules of process plan while minimizing the priority weighted total lateness in genetic algorithm based system from 1000th iteration is given below:

System execution time: 0.391 Seconds

Achievable Priority weighted Total lateness: 61900.00

One of the schedules from final population and its objective value

8	11	18	11	24	35	2	20	11	33	5	20
34	24	36	8	11	20	10	25	34	1	21	14
25	35	7	13	17	13	23	31	1	18	13	35
26	2	18	26	3	17	23	7	15	33	3	17
7	15	34	8	13	16	14	26	30	7	10	17
13	23	29	6	11	20	13	24	27	1	19	14
36	5	17	25	10	2	15	21	26	10	36	6
21	13	34	5	16	23	1	18	24	34	2	21
22	35	4	16	24	6	17	10	1	19	22	35
7	7	16	7	22	27	8	15	17	13	23	29
5	20	26	29	4	21	24	31	2	19	13	24
2	17	15	35	23	5	18	22	6	8	10	19
22	15	31	3	8	14	17	24	13	35	1	18
13	4	16	7	30	5	19	22	7	31	3	16
24	10	34	5	21	26	13	35	5	19	15	2
19	15	1	14	19	15	35	25	10	27	12	32
18	11	29	20	36	32	31	32	33	33	33	37
26	23	14	23	9	6	14	17	15	23	34	11
9	24	10	32	5	9	11	37	Objective value: 61900			

One of schedule obtained by minimizing the priority weighted total flow time by simulated annealing algorithm based system is given below:

System execution time: 0.100 Seconds

Achievable priority weighted total flow time: 46081.00

Schedule which achieves this priority weighted total flow time

8	14	18	10	25	35	2	20	12	34	2	20
34	24	36	7	15	21	14	23	36	4	21	10
26	34	7	14	17	14	22	31	5	16	12	35
25	2	19	24	5	18	25	8	12	35	5	19
8	13	30	8	14	20	15	26	31	7	11	16
13	22	31	2	12	19	10	23	27	5	18	12
36	2	19	23	14	6	11	21	25	14	36	1
19	11	35	4	17	24	1	17	24	35	4	19
25	36	1	19	22	6	18	7	3	17	24	34
7	7	16	7	22	27	8	10	19	15	22	31
2	17	26	36	5	17	25	30	3	18	12	23
1	18	14	35	23	5	17	24	3	8	14	19
26	12	29	1	8	14	17	25	15	35	4	16
12	2	17	8	30	2	18	22	7	27	1	17
22	13	35	6	18	24	14	34	2	20	12	4
21	10	5	15	19	12	35	22	15	28	14	29
17	14	29	21	36	32	32	32	33	33	33	37
24	23	11	23	9	4	11	16	13	22	35	11
9	23	11	33	6	9	14	35				