CHAPTER 2

LITERATURE REVIEW

2.1 INTRODUCTION

Scheduling is concerned with the optimal allocation of scarce resources to activities over time. It has been the subject of extensive research since the early 1950s. Much of the early work on scheduling was concerned with the analysis of single machine systems. Later more complex scheduling systems were investigated. General survey papers on sequencing and scheduling by Lawler et al (1993), Anderson et al (1997), Chung-Yee Lee (1997) and books written by different authors viz. Brucker (1995), Pinedo (1995) and Blazewicz et al (1996) are also available.

Scheduling covers a wide range of problems and concerns all industrial systems. The characteristics of scheduling problems are various and numerous, and their modelling is a difficult task.

In general, according to Garey and Johnson (1979) most of the scheduling problems are NP-hard, consequently there are no known algorithms guaranteed to give an optimal solution and run in polynomial time. This has led to a long history of techniques emanating from the fields of artificial intelligence and operations research that provide approximate solutions to fairly general classes of problems or exact solutions to highly
specific and restricted problems. This tends to rely on the use of heuristics or some form of stochastic optimization technique or a mixture of both.

A detailed theoretical analysis of the scheduling problem is given by Garey and Johnson (1979). Well known examples of traditional approaches to solve scheduling problems are described in Balas (1969) and Carlier and Pinson (1989). General coverage of scheduling can be found in Muth and Thompson (1963) and French (1982).

2.2 HEURISTICS FOR THE FLOWSHOP SCHEDULING PROBLEM

The complexity of the flowshop scheduling problem renders exact solution methods that are impractical for instances of more than a few jobs and / or machines. This is the main reason for the various heuristic methods proposed in the literature. The heuristics can be divided into constructive heuristics and improvement heuristics. Constructive heuristics are used to build a feasible schedule from scratch. The improvement heuristics are used to improve a previously generated schedule by normally applying some form of specific problem knowledge.

2.2.1 Constructive Heuristics

Johnson’s algorithm (1954) is the earliest known heuristic for the flowshop scheduling problem, which provides an optimal solution for two machines. Moreover, it can be used as a heuristic for the m machine case by clustering the m machines into two “virtual” machines. Other authors have used the general ideas of Johnson’s rule in their algorithms. For example, Dudex and Teuton (1964) developed an m-stage rule for the Permutation Flowshop Scheduling Problem (PFSP) that minimizes the idle time
accumulated on the last machine when processing each job by using Johnson’s approach.

Another approach by Palmer (1965) is to assign “weight” or “index” to every job and then arrange sequence by sorting the jobs according to the assigned index. This idea was first exploited by Palmer (1965) when he developed a very simple heuristic in which for every job a “slope index” was calculated and then the jobs are scheduled by non-increasing order of this index.

Campbell et al (1970) developed a heuristic by extending Johnson’s algorithm. This heuristic is known as CDS and builds m – 1 schedules by clustering the m original machines into two virtual machines and solving the generated two machine problem by repeatedly using Johnson’s rule.

Gupta (1971) proposed a modification of Palmer’s slope index which exploited some similarities between scheduling and sorting problems. Similarly, Bonney and Gundry (1976) worked on the idea of using the geometrical properties of the cumulative process times of the jobs and a slope matching method for scheduling the PFSP.

Dannenbring (1977)’s Rapid Access (RA) heuristic was a mixture of the previous ideas of Johnson’s algorithm and Palmer’s slope index. In this case a virtual two machine problem was defined as in the CDS heuristic, but instead of directly applying Johnson’s algorithm over the processing times, two weighting schemes are calculated, one for each machine, and then Johnson’s algorithm is applied. The weighting schemes give the processing times for the jobs in the two virtual machines. Since all the jobs in a PFSP form a permutation, many proposed methods work with the idea of
exchanging the position the jobs in the sequence or inserting jobs at different locations to obtain better results.

Nawaz et al (1983) proposed NEH heuristic for PFSP to minimize makespan. It is based on the idea that jobs with high processing times on all the machines should be scheduled in the sequence as early as possible. Hence, the NEH heuristic is based neither on Johnson’s algorithm nor on slope indexes. The only drawback is that a total of \( \frac{n \times (n+1)}{2} - 1 \) schedules have to be evaluated, being n of those schedules complete sequences.

Hundal and Rajgopal (1988) proposed a very simple extension to Palmer’s heuristic by exploiting the fact that when \( m \) is an odd number, Palmer’s slope index returns the value 0 for the machine \( (m + 1)/2 \) and consequently ignores it. In order to overcome this, two more slope indexes are calculated and with these two slope indexes and the original Palmer’s slope index, three schedules are calculated and the best one is given as a result.

Sarin and Lefoka (1993) exploited the idea of minimizing idle time on the last machine since any increase in the idle time on the last machine will translate into an increase in the total completion time or makespan. In this way, the sequence is completed by inserting one job at a time and priority is given to the job that, once added to the sequence, would result in minimal added idle time on machine \( m \). The method compares well with the NEH heuristic, only when the number of machines in a problem exceeds the number of jobs.

Koulamas (1998) reported a two phase heuristic, called HFC. In the first phase, the HFC heuristic makes extensive use of Johnson’s algorithm. The second phase improves the resulting schedule from the first phase by
allowing job passing between machines, i.e. by allowing non-permutation schedules.

Davoud Pour (2001) proposed another insertion method. This new heuristic is based on the idea of job exchanging and is similar to the NEH method. Framinan et al (2003) have published a study about the NEH heuristic where different initialisations and orderings are considered. The study also includes different objective functions.

### 2.2.2 Improvement Heuristics

Contrary to constructive heuristics, improvement heuristics start from an already built schedule and try to improve it by some given procedure. Ho and Chang (1991) developed a method that works with the idea of minimising the elapsed times between the end of the processing of a job in a machine and the beginning of the processing of the same job in the following machine in the sequence. The authors refer to this time as “gap”. The algorithm calculates the gaps for every possible pair of jobs and machines and then by a series of calculations, the heuristic swaps jobs depending on the value of the gaps associated with them. The heuristic starts from the CDS heuristic by Campbell et al (1970).

Suliman (2000) developed an improvement heuristic. In the first phase, a schedule is generated with the CDS heuristic method. In the second phase, the schedule generated is improved with a job pair exchange mechanism, a directionality constraint is imposed to reduce the search space. For example, by moving a job forward, a better schedule is obtained. It is assumed that better schedules can be achieved by maintaining the forward movement and not allowing a backward movement. Ponnambalam et al (2001) has compared five different heuristics against 21 typical test problems.
2.2.3 Metaheuristics

There is plenty of research work done on application of metaheuristics for PFSP. In this section, we will point out a few noteworthy papers mainly dealing with Simulated Annealing (SA), Genetic Algorithms (GA) and other metaheuristics, as well as hybrid methods.

Osman and Potts (1989) proposed a simple Simulated Annealing algorithm using a shift neighbourhood and a random neighbourhood search. Widmer and Hertz (1989) presented a method called SPIRIT, which is a two phase heuristic. In the first phase, a problem is generated with an analogy with the Open Travelling Salesman Problem (OTSP) and then, this problem is solved with an insertion method to obtain an initial solution. In the second phase, a Tabu Search metaheuristic with standard parameters and exchange neighbourhood is used to improve the incumbent solution. Taillard (1990) also presented a similar procedure to that of Widmer and Hertz. A Tabu Search technique is applied to a schedule generated by an improved NEH heuristic. Taillard tested various types of neighbourhoods and the neighbourhood resulting from changing the position of one job proved to be the best.

Oggu and Smith (1990a) proposed a Simulated Annealing approach to the PFSP which involved an initialisation with the Palmer and Dannenbring heuristics. The choice of a large neighbourhood and an acceptance probability function independent of the change in the makespan of the schedule resulted in near-optimal schedules for the problems tested. In a later work (Oggu and Smith, 1990b) this Simulated Annealing is compared with that of Osman and Potts (1989). The results show a tie between the two metaheuristics with a slight advantage on the part of Osman and Potts’s one.
Werner (1993) constructed a fast iterative method which obtained very good results by generating restricted numbers of paths in a search neighbourhood built upon the path structure of a feasible schedule. It is interesting to remark that while local search or fast iterative methods operate in a very small neighbourhood, Werner’s path algorithm operates with a rather large neighbourhood but only evaluates the most interesting solutions in each step.

Reeves (1993) modified the SPIRIT algorithm and introduced several enhancements, such as the initialisation by the NEH heuristic and the insertion neighbourhood. The author compared this algorithm with the SA of Osman and Potts (1989) showing better results.

Chen et al (1995) developed a simple Genetic Algorithm for the PFSP with various enhancements. The initial population is generated with the CDS and RA heuristics and also from simple job exchanges of some of the individuals. Only the crossover operator is applied (no mutation), and the crossover used is the Partially mapped crossover or PMX. Reeves (1995) also developed a Genetic Algorithm. In this case, the offspring generated in each step of the algorithm do not replace their parents but individuals from the generation that have a fitness value below average. Reeves used a crossover operator called C1, which is equivalent to the One Point Order Crossover. Another remarkable feature of the algorithm is that it uses an adaptive mutation rate. The algorithm uses a shift mutation which simply changes the position of one job. Reeves also chose to seed the initial population with a good sequence among randomly generated ones. This good sequence is obtained with the NEH heuristic. Also, the selection of the parents is somewhat different from what is common in genetic algorithms; parent 1 is selected using a fitness rank distribution whereas parent 2 is chosen using a uniform distribution.
Ishibuchi et al (1995) presented two Simulated Annealing algorithms characterized by having robust performance with respect to the temperature cooling schedule. Their results showed that their algorithms were comparable to the SA of Osman and Potts (1989). Zegordi et al (1995) demonstrated a hybrid technique by introducing problem domain knowledge into a Simulated Annealing algorithm. Their algorithm, called SA-MDJ, uses a “Move desirability for Jobs” table which incorporates several rules that facilitate the annealing process. The algorithm is compared with Osman and Potts’s SA algorithm proving to be slightly inferior but much faster in the instances tested.

The Tabu Search of Moccellin (1995) is mainly based on Widmer and Hertz’s SPIRIT heuristic. The only difference exists in the step of calculating the initial solution for the problem. The analogy with the Travelling Salesman Problem is maintained, but the way the distance between jobs is calculated differs from that of Widmer and Hertz. Also, how the TSP problem is solved is different, since Moccellin uses the farthest Insertion Travelling Salesman Procedure (FITSP). The rest of the algorithm is essentially similar to SPIRIT.

The genetic algorithm proposed by Murata et al (1996a) uses the two-point crossover operator and a shift mutation along with an elitist strategy to obtain good solutions for the PFSP. The algorithm performed worse than implementations of Tabu Search, Simulated Annealing and local search, so the authors implemented two hybrid versions of the Genetic Algorithm; Genetic Simulated Annealing and Genetic Local Search. In these algorithms, an “improvement” phase is performed before selection and crossover and this improvement is made with local search and simulated annealing algorithms respectively. The hybrid algorithms performed better than the non hybrid
Genetic Algorithm, implementations of Tabu Search, Simulated Annealing and local search.

Nowicki and Smutnicki (1996) proposed a Tabu Search metaheuristic where once again, a reduced part of the neighbourhood is evaluated along with a fast method for obtaining the makespan. The neighbourhood is reduced with the idea of blocks of jobs, where jobs are clustered and the movements are made on block-by-block basis instead of just moving single jobs.

Another hybrid Genetic Algorithm is that of Reeves and Yamada (1998), the idea behind this algorithm is the Multi-step crossover fusion or MSXF operator which coalesces a crossover operator with local search. The MSXF operator carries a biased local search from one parent using the other parent as a reference. The MSXF operator is defined by a neighbourhood structure and a measure of distance, which has some similarities with simulated annealing but with a constant temperature parameter. Another metaheuristic is given by Stutzle (1998), and is given the name “Iterated Local Search” or ILS. Somehow or other, the ideas behind ILS are those of descent search, local search, Simulated Annealing or Tabu Search; an initial solution is obtained and a LocalSearch procedure is executed over this initial solution, then a Modify procedure that slightly modifies the current solution is carried out. Then, LocalSearch is called again and the resulting schedule is deemed as the new current solution only if a certain acceptance criterion is met (procedure AcceptanceCriterion). The ILS procedure may also work with a list called history that works similarly to the tabu list in Tabu Search. According to the tests conducted by Stutzle, the ILS algorithm is much better than the Tabu Search of Taillard (1990) and also better than the Tabu Search of Nowicki and Smutnicki (1996).
Ben-Daya and Al-Fawzan (1998) implemented a Tabu Search algorithm with some extra features such as intensification and diversification schemes that provide better moves in the tabu search process. The algorithm proposed provides similar results as the TS of Taillard and slightly better results than the SA of Ogbu and Smith. Moccellin and dos Santos (2000) presented a hybrid Tabu Search-Simulated Annealing heuristic that is compared with simple Tabu Search and simple Simulated Annealing implementations from the same authors, showing advantages for the hybrid approach.

Ponnambalam et al (2001) evaluate a Genetic Algorithm which uses the GPX crossover (Generalised Position Crossover) and other features like shift mutation and a randomized initial solution.

Wodecki and Bozejko (2002) have proposed a SA algorithm that is run in a parallel computing environment. The authors compare the proposed algorithm with the NEH heuristic, the former showing better results.

Wang and Zheng (2003a) incorporated the NEH heuristic into random initialization of a GA and proposed effective hybrid heuristic for flowshop scheduling and pointed out that their hybrid heuristic was comparable or even superior to that of Tabu search, simulated annealing and GA. Wang and Zheng (2003b) uses modified evolutionary programming (MEP) and they showed that their MEP was superior to the simple evolutionary programming and NEH heuristic (Nawaz et al 1983) method.

Reza Hejazi and Saghaian (2005) gave a complete survey of flowshop-scheduling problems and contributions from early works of Johnson of 1954 to recent approaches of metaheuristics. They mainly considers a flowshop problem with a makespan criterion and it surveys some exact
methods (for small size problems), constructive heuristics and developed improving metaheuristic and evolutionary approaches as well as some well-known properties and rules for flowshop problem.

Liang Zhang et al (2006) proposed Adaptive Genetic Algorithm (AGA) for flowshop scheduling problem with makespan objective. They have investigated adaptation and hybridisation of genetic operators and proposes AGA, where the utilising ratios of multiple genetic operators are adaptively and dynamically controlled according to their contribution to the searching process.

Noorul Haq et al (2007) have proposed Scatter Search algorithm for scheduling problems of a flowshop with makespan objective.

Table 2.1 shows a summary of the different constructive and improvement heuristics reviewed in chronological order, and Table 2.2 shows information about the reviewed metaheuristics.
<table>
<thead>
<tr>
<th>Year</th>
<th>Author/s</th>
<th>Heuristic</th>
<th>Type*</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1954</td>
<td>Johnson</td>
<td>John</td>
<td>C</td>
<td>exact for two machine case</td>
</tr>
<tr>
<td>1964</td>
<td>Dudex and Teuton</td>
<td>John</td>
<td>C</td>
<td>based on Johnson’s rule</td>
</tr>
<tr>
<td>1965</td>
<td>Palmer</td>
<td>Palme</td>
<td>C</td>
<td>based on slope indexes</td>
</tr>
<tr>
<td>1970</td>
<td>Campbell et al</td>
<td>CDS</td>
<td>C</td>
<td>based on Johnson’s rule</td>
</tr>
<tr>
<td>1971</td>
<td>Gupta</td>
<td>Gupta</td>
<td>C</td>
<td>based on slope indexes</td>
</tr>
<tr>
<td>1972</td>
<td>Gupta</td>
<td></td>
<td>C</td>
<td>three heuristics considered</td>
</tr>
<tr>
<td>1976</td>
<td>Bonney and Gundry</td>
<td>RA, RACS, RAES</td>
<td>C/I</td>
<td>three heuristics considered: RA, RACS and RAES</td>
</tr>
<tr>
<td>1983</td>
<td>Nawaz et al</td>
<td>NEH</td>
<td>C</td>
<td>job priority / insertion</td>
</tr>
<tr>
<td>1988</td>
<td>Hundal and Rajagopal</td>
<td>HunRa</td>
<td>C</td>
<td>Palmer’s based</td>
</tr>
<tr>
<td>1991</td>
<td>Ho and Chang</td>
<td>HoCha</td>
<td>I</td>
<td>Between-jobs gap minimization</td>
</tr>
<tr>
<td>1993</td>
<td>Sarin and Lefoka</td>
<td></td>
<td>C</td>
<td>last machine idle time minimisation</td>
</tr>
<tr>
<td>1995</td>
<td>Rajendran</td>
<td>CR(MC)</td>
<td>C</td>
<td>Initial seed sequence from CDS algorithm</td>
</tr>
<tr>
<td>1998</td>
<td>Koulamas</td>
<td>Koula</td>
<td>C/I</td>
<td>two phases, 1st phase improvement by job passing</td>
</tr>
<tr>
<td>2000</td>
<td>Suliman</td>
<td>Sulin</td>
<td>I</td>
<td>job pair exchange</td>
</tr>
<tr>
<td>2001</td>
<td>Davoud Pour</td>
<td>Pour</td>
<td>C</td>
<td>job exchanging</td>
</tr>
<tr>
<td>2003</td>
<td>Framinan et al</td>
<td></td>
<td>C</td>
<td>study on the NEH heuristic</td>
</tr>
<tr>
<td>2005</td>
<td>Ravindran et al</td>
<td>HAMC</td>
<td>I</td>
<td>Heuristic used CR(MC)</td>
</tr>
</tbody>
</table>

* C: Constructive, I: Improvement
<table>
<thead>
<tr>
<th>Year</th>
<th>Author/s</th>
<th>Acronym</th>
<th>Type</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>1989</td>
<td>Osman and Potts Widmer and Hertz</td>
<td>SAOP</td>
<td>SA</td>
<td>Initial solution based on the OTSP</td>
</tr>
<tr>
<td>1990</td>
<td>Taillard Ogbu and Smith</td>
<td>Spirit</td>
<td>TS</td>
<td></td>
</tr>
<tr>
<td>1990</td>
<td>Werner Reeves</td>
<td>TS</td>
<td>TS</td>
<td></td>
</tr>
<tr>
<td>1993</td>
<td>Werner Reeves</td>
<td>TS</td>
<td>TS</td>
<td></td>
</tr>
<tr>
<td>1995</td>
<td>Chen et al Reeves Ishibuchi et al Zegordi et al Moccellin</td>
<td>GACHen GAREev</td>
<td>GA</td>
<td>PMX crossover adaptive mutation rate two SA considered Combines sequence knowledge based on SPIRIT</td>
</tr>
<tr>
<td>1996</td>
<td>Murata et al Nowicki and Smutnicki</td>
<td>GAMIT</td>
<td>Hybrid</td>
<td>GA + Local Search / SA neighbourhood by blocks of jobs</td>
</tr>
<tr>
<td>1998</td>
<td>Stutzle Ben-Daya and Al-Fawzan Reeves and Yamada</td>
<td>ILS</td>
<td>other</td>
<td>Iterated Local Search intensification + diversification GA operators with problem knowledge</td>
</tr>
<tr>
<td>2000</td>
<td>Moccellin and dos Santos</td>
<td>Hybrid</td>
<td>TS</td>
<td>TS + SA</td>
</tr>
<tr>
<td>2001</td>
<td>Ponnambalam et al Wodecki and Bozejko</td>
<td>GAPAC</td>
<td>GA</td>
<td>GPX crossover parallel simulated annealing</td>
</tr>
<tr>
<td>2003</td>
<td>Wang and Zheng</td>
<td>Hybrid</td>
<td>GA</td>
<td>GA + SA, multicrossover operators</td>
</tr>
<tr>
<td>2004</td>
<td>Ponnambalam et al</td>
<td>TSPGA</td>
<td>GA</td>
<td>TSP algorithm with GA for multi-objective PFSP</td>
</tr>
<tr>
<td>2007</td>
<td>Noorul Haq et al</td>
<td>SS</td>
<td>SS</td>
<td>Scatter Search</td>
</tr>
</tbody>
</table>
During the last decade, GA and SA have been widely applied to many engineering fields, especially in the production scheduling field (Wang and Zheng 2003a). GA exhibit parallelisms contain certain redundant and historical information of the past solutions and are suited to implementation for large parallel architectures (Goldberg 1989). However, it is not easy to regulate the convergence of GA and to choose suitable parameters and operators, so GA often suffer from premature convergence (Leung et al 1997). A large amount of work has been done to enhance the performance of GA.

In this thesis an attempt is made to improve the existing genetic algorithm and simulated annealing algorithm procedure and applies them in permutation flowshop scheduling to get better results.

2.3 FLOWSHOP SCHEDULING WITH TOTAL (OR MEAN) FLOWTIME

Many multi-stage production systems can be modelled as flow shops. Therefore, flowshop sequencing problems have been extensively studied. Most of the studies try to minimize the makespan. In today’s highly competitive global market, quick response to customer orders becomes increasingly important. This requires us to use more relevant criteria in making scheduling decisions. An appropriate criterion is to minimize the mean flowtime (or total flowtime). Compared to the Cmax problem, the Csum problem is more difficult to optimize, mainly because the calculation of the objective function is more time consuming (Takeshi Yamada and Reeves 1998).

Some of the heuristic procedures for minimisation of makespan have been developed by Campbell et al (1970), Dannenbring (1977), Nawaz
et al (1983), Widmer and Hertz (1989), Osman and Potts (1989), Taillard (1990), Ogbu and Smith (1990), Ishibuchi et al (1995), and Ben-Daya and Al-Fawzan (1998). While all these heuristics seek to minimize makespan, heuristics have also been proposed to minimize total or mean flowtime (Ho 1995, Miyazaki et al 1978, Chandrasekharan Rajendran and Hans Ziegler 1999). For the problem of scheduling in a flowline-based manufacturing cell with missing operations for jobs in a part-family, Logendran and Nudtasomboon (1991) and Sridhar and Rajendran (1993) have proposed heuristics to minimize makespan and total flowtime, respectively. The objective of scheduling to minimize total flowtime is significant in many real-life situations, especially with respect to the minimization of inventory or holding costs (Baker 1974 and French, 1982).

Xixng Gao et al (2005) have proposed Double Inserting Heuristic for flow shops with total flowtime minimization, in which the initial solution is improved by combing local insertion with global insertion.

### 2.4 Multi-Objective Flowshop Scheduling

The majority of works on flowshop problem studies the problem in its single criterion form and aims mainly to minimize makespan or total (mean) flowtime. However, the desirability of a schedule being evaluated by more than one performance measure is often cited in the literature. Apart from the makespan and total flowtime objectives, another significant objective in flowshop scheduling is the minimization of total machine idletime. Ruiz-Diaz and French (1983) provide one of the earliest survey papers on multi-objective scheduling. As seen early, Ho and Chang (1991) attempts to minimize not only makespan, but also total flowtime and machine idletime. A survey of the flowshop scheduling literature has revealed that there is hardly
any research work carried out with multiple criteria such as the minimization of makespan, mean flowtime and machine idletime.

Many real-world decision-making problems involve simultaneous optimization of multiple objectives (Deb 1998). Real life scheduling problems require the decision maker to consider a number of criteria before arriving at any decision. Thus considering problems with more than one criterion is more relevant in the context of real life scheduling problems. Several bi-objective approaches exist in the literature. Rajendran (1995) proposed a specific heuristic to minimize makespan, total flowtime and machine idletime. Amit Nagar et al (1995) gave a survey of the existing multicriteria approaches of scheduling problems.

2.4.1 Multiobjective Metaheuristics

Several multiobjective metaheuristics using local search have been put forward in the literature. Some of these multiobjective metaheuristics are briefly described below:

Jagabandhu Sridhar and Chandrasekharan Rajendran (1996) have proposed a genetic algorithm to solve the problem of scheduling in flowshop and flowline based cellular manufacturing systems by considering the objectives of minimizing makespan, total flowtime and machine idletime.

Ponnambalam et al (2001) have proposed Multi objective genetic algorithm to solve job shop scheduling problems. The performance criterion considered was the weighted sum of the multiple objectives minimization of makespan, minimization of total idletime of machines and minimization of total tardiness.
Ponnambalam and Mohan Reddy (2003) have proposed a multiobjective hybrid evolutionary search algorithm which combines a genetic algorithm and a simulated annealing algorithm. This algorithm was used in flow-line environment for minimizing makespan, overtime and holding cost.

Vinicius Amaral Armentano and Jose Elias Claudio Arroyo (2004) have proposed a Tabu Search algorithm for multi-objective combinatorial problems with a goal of obtaining a good approximation of the efficient solutions. They applied this algorithm to the permutation flowshop scheduling problem in order to minimize the criteria of makespan and maximum tardiness.

Ravindran et al (2005) considered the multicriterion approach to flowshop scheduling problems by considering makespan and total flowtime. They proposed three heuristics namely HAMC1, HAMC2 and HAMC3 for the above objectives.

Pasupathy et al (2005) have proposed a pareto ranking based multi-objective genetic algorithm to solve flowshop scheduling problems with the objectives of minimizing the makespan and total flowtime of jobs.

2.4.2 Weighted Criteria Approach

In solving multi-objective optimization problems, one of the popular approaches is to convert the multi-criteria into a single objective using a weighted sum of the criteria, and then solving the problem as a single criterion problem (Venkata Ranga Neppalli 1996 and Murata et al 1996b). The weight assigning aspect of this approach is vital and usually projects the relative importance of each criterion.
Venkata Ranga Neppalli et al (1996) used the Vector Evaluated Approach (VEA) and Weighted Criteria Approach (WCA) based GA algorithms to solve the same set of problems. He stated that the computational results of both the Vector Evaluated Approach and Weighted Criteria Approach are effective in solving the two-stage bicriteria flowshop problem. Both the approaches are equally good. Also they stated that, the difference in performance for VEA and WCA is not significant and hence, either of the two approaches can be used to measure the fitness of a solution in the population without affecting the performance of the GA based approach.

Murata et al (1996b) proposed a multiobjective genetic algorithm for the flow shop problem with two and three objectives. The criteria considered were total makespan, total tardiness and total flow time. Their genetic algorithm used weighted vectors generated at random in each selection step. That is, before selection, a vector of weights is generated at random and all the individuals in the population are evaluated using that vector. The weights are used in linear scalarizing functions. The fitness of solutions changes from iteration to iteration and depends on the value of the current weighted linear scalarizing function. They found that their approach with variable weights was capable of approximating the Pareto optimal set in non-convex fronts and produced better results than the VEA.

Ponnambalam et al (2004) have proposed a multi-objective evolutionary search algorithm using a traveling salesman algorithm and genetic algorithm for flowshop scheduling problem. In this algorithm initial sequence is obtained by solving the TSP. The initial population of the genetic algorithm is created with the help of a neighbourhood creation scheme known as a random insertion perturbation scheme, which uses the sequence obtained from TSP. Also this algorithm used a weighted sum of multiple objectives as
a fitness function. The weights were randomly generated for each generation to enable a multi-directional search. The performance measures considered include minimising makespan, mean flow time and machine idle time.

2.5 RESEARCH GAP

- Flow shop scheduling is a class of widely studied scheduling problems with a strong engineering background, which illustrates at least some of the demands required by a wide range of real-world problems and has earned a reputation for being difficult to solve (Garey et al 1976 and Pinedo 2002).

- Search-based algorithms have been widely used for solving the permutation flowshop problems (Ruben Ruiz et al 2003 and Wang & Zheng 2003).

- By suitably selecting the parameters used in search heuristics, there is a scope to improve the results already reported.

- Desirability of a schedule evaluated by more than one performance measure is often cited in the literature (Murata et al 1996b and Hisao Ishibuchi et al 2003).

- A survey of the flowshop scheduling literature has revealed that there is hardly any research work carried out with multiple criteria such as the minimization of makespan, mean flowtime and machine idle time.
2.6 OBJECTIVES OF THE THESIS

- To develop a heuristic model for scheduling permutation flow shop problem.
- To modify the existing heuristics to improve the solution quality.
- To extend the heuristics to handle multi-objective flow shop scheduling problems.

2.7 SCOPE OF THE THESIS

- To study and analyze the different meta-heuristic methods like Genetic Algorithm and Simulated Annealing algorithm for solving the permutation flowshop scheduling problems.
- To find some better method of solving the flowshop scheduling problems.
- To develop Improved Genetic Algorithm (IGA) and modified Simulated Annealing algorithm (SAA) to solve flowshop scheduling problem with the objective of minimizing makespan.
- Extending the IGA and modified SAA to solve flowshop scheduling problem with the objective of minimizing total (or mean) flowtime of all jobs.
- Extending the IGA to solve flowshop scheduling problem with Bi-criteria objective of minimizing makespan and total flowtime.

- Extending the Improved Genetic Algorithm (IGA) to solve flowshop scheduling problem with Multi-objective of minimizing makespan, total flowtime and total machine idletime.

2.8 SUMMARY

A comprehensive review and evaluation of many existing heuristics and metaheuristics for the permutation flowshop with single and multi-objective criterion has been presented in this chapter. The drawbacks and limitations in the current state of art have been identified and direction for the current research has been indicated.