CHAPTER 1

INTRODUCTION

1.1 BACKGROUND

A general flow shop scheduling problem (FSP) with n jobs and m machines is considered, in which each job i, i=1,2,.....,n needs to be processed on each machine j, j=1,2,.....,m, during an uninterrupted processing time t_{ij} ≥ 0. In such cases, minimizing the make span is considered as one of the most important performance measures that has to be optimized. The number of possible sequences can be (n!)^m. Some common performance evaluation objective functions of a FSP are:

- Makespan – total time to completely process all jobs (Most Common)
- Average Time of jobs in the shop
- Lateness
- Average Number of jobs in the shop
- Utilization of machines
- Utilization of workers

The permutation flow shop sequencing problem with the objective of makespan minimization, involves the determination of the order of processing n jobs in m machines. Here, as the order of processing is the same for all the jobs, the possibilities are reduced to simply, n!. The exact approaches for the PFSP under makespan criterion (PFSP-C_{max}) are fairly effective, but only for a small number of jobs, and specially, machines. Solving larger problems becomes extremely difficult/impossible using the exact methods. Hence, heuristic methods are widely used as they can give reasonably good results.

Our objective is to find a processing order of n jobs, the same for each machine, such that the make span is minimized. That is, n jobs are finished as soon as possible. It is assumed that all jobs are available for processing at time zero. At any
time, each machine can process at the most one job and each job can be processed on at the most one machine. The capacity of the queue for each machine is unlimited.

The function grows exponentially with an increase in the problem size. But, for a problem with 2 machines and n jobs, Johnson (1954) had developed a polynomial algorithm to get an optimal sequence (more than one optimal solution may be available for the same problem), that is, in a definite time, one can get an optimal solution. Johnson’s algorithm can be extended to three machines and n jobs problems. For n jobs and m machines, many methods have been proposed over the years.

Most of the studies conclude that the heuristic developed by Nawaz, Enscore and Ham (NEH heuristic, 1983) is the most efficient so far in its category. Framinan et al. (2003) considered twenty two different approaches for the indicator value, and eight different sorting criteria, totaling 176 approaches for every objective function. Additionally, for every objective function, the RANDOM choice of a sequence is considered. The processing times were drawn randomly as integers from a discrete uniform distribution between 1 and 99. It was concluded that for the makespan, the best five-tupel consists of the NEH-insertion approach which selects the first two jobs as the initial sequence.

Though the indicator values and sorting sequences are unlimited, the important ones were covered by Framinan et al. However, till today, many researchers have been trying to improve the NEH Algorithm in their own ways.

As such, most of the heuristics offer one optimal / near optimal solutions for general flow shop scheduling problems. The CDS (1970) heuristic offers a set of sequences (m-1) which contain the optimal and a few more near optimum solutions with less effort. Except for a few, most of the procedures are a bit complicated and require some level of expertise to understand and implement them. At the shop floor level, the requirement is a heuristic which is simple, fast, efficient, and at the same time, should be able to give more than one sequence having optimal or near optimal make spans. If more than one sequence having optimal/ near optimal makespans are
in hand, in the case of any constraint, another sequence can be proposed for processing, without much reduction in the total completion time of all the jobs.

If the general processes are analyzed, it can be observed that they can be conveniently grouped into three categories: processes that require uninterrupted power supply like moulding, forging, casting; processes that require power supply but, break down in supply is permitted in between, like drilling, turning, milling, and processes that do not require power supply and can be continued during power breakdown time also, like manual cleaning, and assembly. On many occasions, for a known time span, production may not take place owing to many reasons.

Any scheduling problem essentially depends upon three important factors, namely, job transportation time which includes moving time and idle time, the relative importance of a job over another, and breakdown machine time (www.euroasiapub.org). These three factors were separately studied by many researchers. Most of the optimization criteria are based on the completion times of the jobs at the different machines (www.upv.es). Also, many methods consider the processing time in different forms for reaching a solution. In addition to the processing time, the time taken for a job to move from one machine after processing to another machine for processing, is also significant on many occasions. This condition has been studied by a few researchers in the past.

A few researchers have proposed to check a set of structural conditions to combine the processing times and jobs transportation times, to convert the three machine (in fact, a total of 5 sets of readings equivalent to 5 machines) problem in to a two machine problem, and subsequently solved it using Johnson’s algorithm. After checking the conditions, the time units were combined in a specific pattern, to obtain the processing times of two hypothetical machines. It may be noted that, the CDS Algorithm takes care of the first and last conditions, while evaluating (m-1) cases to obtain an optimal make span.

Influenced by the above observations, it has been decided to take up a study of the permutation flow shop scheduling problems, with the objective of makespan
optimization. It is expected that the outcome of this work will be useful for industrial engineers in scheduling jobs in an optimum way. The simple proposed methods and models for improving the makespan and computing the sequences can be used by the shop floor supervisors with a little effort.

1.2 SEQUENCING AND SCHEDULING

Sequencing and scheduling are two important words that are being used in a manufacturing flow shop environment. They exist whenever there is more than one possibility in which a number of tasks can be performed. Sequencing is a technique, to order the jobs for processing in a particular sequence, whereas, scheduling refers to the allocation of resources over time to perform a section of tasks. There are different types of sequencing, which are followed in industries. The sequence which gives the minimum processing time is adopted. Scheduling means, allocation of resources over time to perform a section of tasks.

1.3 SCHEDULING ENVIRONMENTS

Sequencing and scheduling can have the following environments:

- One machine shop- single machine has to process n number of jobs.

- Flow shop- there are m machines available in series for processing n jobs. They can be processed either by: Permutational - all the jobs will be processed in the series of m machines in the same order; Non- Permutational- all the jobs need not be processed in the same order in the series of m machines.

- Job shop- each job has its own flow pattern and the subsets of these jobs can visit the machines more than once. That is, multiple entries and exits are permitted here.

- Assembly job shop- the jobs that are being processed will be having at least two components and one assembly operation.

- Hybrid job shop- the precedence ordering of the operations of a few jobs are the same.
• Open shop- there is no specific flow pattern for the jobs. There is no restriction in the routing in processing the n jobs in m machines.

• Closed shop- it is a type of job shop. All the production orders are generated as a result of inventory replenishment decisions.

1.4 ASSUMPTIONS

While solving the problems, it is required to have a few assumptions in any sequencing and scheduling environment. They may vary depending on the ground situations. They are presented here for reference.

1. The sets of jobs and machines are known in advance and fixed.

2. All the jobs and machines are available at time zero and are independent.

3. An infinite in-process storage buffer is assumed. If a given job needs an unavailable machine then it joins a queue of unlimited size waiting for that machine.

4. The processing times are fixed for all machines, have a known probability distribution function, and are sequence independent

5. Set up times are included in the processing times.

6. All jobs and machines are equally weighed.

7. No pre-emption is allowed (for a general flow shop problem).

8. A definite due date is assigned to each job.

9. Each job is processed in the assigned machines and each machine processes the assigned jobs.

10. Each machine can process at the most one job at a time and each job can be processed only on one machine at a time.

11. The process plan is known in advance and is fixed.
1.5 METHODS OF SOLUTIONS

Several methods have been developed in the past for solving sequencing and scheduling problems. They belong to the four categories of deterministic, static, dynamic and stochastic methods.

Deterministic- when all the elements of the given problem like the state of arrival to the machine, due dates of jobs, processing times in different machines, and the machines’ availability data do not include any stochastic factors (that have a probability). They can be very well determined in advance.

Static- the sets of jobs do not change over time and are available before hand. Except this nature, others conditions are similar to those of the deterministic category.

Dynamic- jobs arrive at different times. The total quantities of jobs do not remain constant in the system over time.

Stochastic- at least one of the problem elements has a stochastic factor.

These solution methods can be classified in to the following types:

1. Efficient optimal methods such as Johnson’s rule.
2. Enumerative methods such as branch and bound algorithms.
3. Heuristic methods such as Palmer’s slope index algorithm and CDS algorithm for solving n jobs and m machines problems.
4. Mathematical models like integer programming for solving flow shop scheduling problems with or without constraints.
5. Meta heuristics such as simulated annealing, and Tabu search methods.
6. Evolutionary algorithms like ant colony optimization, particle swam optimization, genetic algorithms.
7. Simulation methods using tailor made software packages.
8. Analytical methods.
1.6 PROBLEM STATEMENT

A flow shop scheduling problem (FSP) with m – machine permutation is considered in which, each job i, i=1,2,…..,n needs to be processed on each machine j, j=1,2,…..,m, in that order during an uninterrupted processing time \( t_{ij} \geq 0 \). The problem is to find a processing order of n jobs, the same for each machine, such that the makespan is minimized; that is, the n jobs are finished as soon as possible. It is assumed that all jobs are available for processing at time zero. At any time, each machine can process at the most one job and each job can be processed on at the most one machine. The capacity of the queue for each machine is unlimited.

If m=2, Johnson’s algorithm generates an optimal solution for problem FSP. If m=3, the extended Johnson’s algorithm can be used subject to the condition of minimum machining time of any job in machine I (OR) minimum machining time of any job in machine III < maximum machining time of any job in machine II.

For, m > 2, various heuristic procedures have been suggested over the years. The number of possible schedules of the flow-shop scheduling problem involving n jobs and m processing centre is \((n!)^m\). If the order in which all jobs are to be processed in each centre is not to be changed, the number of possible schedules of the flow-shop scheduling problem involving n jobs and m centre is \((n!)\). The optimal solution to the problem is to find the sequence of jobs on each centre, in order to complete all the jobs at all the centres in the minimum total time, provided each job is processed on centres 1, 2, 3, ..., m in that order. The general flow-shop scheduling problem is NP-hard.

1.7 MAKESPAN OBJECTIVE

The permutation flow shop sequencing problem with the objective of makespan (Total completion time) minimization:
Characterized as \( Fm|prmu|Cmax \)
The objective function:

Minimize: \( C_{max} \); Subject to, \( C_{max} \geq C_{max}^* \)

The optimal makespan, \( C_{max}^* \geq LB \)

Lower Bound (Minimum possible completion time), \( LB = \max, j (b_j + T_j + a_j) \)

- \( b_j \) - minimum amount of time before machine ‘j’ starts to work
- \( a_j \) - minimum amount of time that it remains inactive after its work up to the end of the operations

- \( T_j \) - total processing time of ‘n’ jobs in ‘m’ machines

\( p_{i,j} \) - processing times of job \( i \) on machine \( j \) (1 \( \leq i \leq n, \ 1 \leq j \leq m) \)

\( b_j = \min, i(\sum_{k=1}^{i-1} p_{i,k}) \)

\( a_j = \min, i(\sum_{k=j+1}^{m} p_{i,k}) \)

\( T_j = (\sum_{i=1}^{n} p_{i,k}) \)

1.8 ORGANIZATION OF THE THESIS

Chapter 2 that follows the ‘Introduction’ presents a comprehensive survey of the literature survey on solving flow shop scheduling problems. Other chapters that follow discuss in detail about the ‘Development of Heuristics and Algorithms’, ‘Development of Two Models with Constraints’ and ‘Results and Discussion’. Applying the developed models, one case study has been carried out in a Mixer Grinder assembly line. Finally, it concludes with the ‘Conclusion, Limitations and Future Scope of Work’.