B. TECH. PROJECT REPORT

On

To Develop an Online Production Scheduling Tool

BY

A.Harshita

DISCIPLINE OF MECHANICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

To Develop an Online Production Scheduling Tool

A PROJECT REPORT

Submitted in partial fulfillment of the
requirements for the award of the degrees

of
BACHELOR OF TECHNOLOGY
in

MECHANICAL ENGINEERING

Submitted by:
A. Harshita

Guided by:
Dr. Bhupesh K Lad
Associate Professor
IIT Indore

INDIAN INSTITUTE OF TECHNOLOGY INDORE
November 2019

CANDIDATE'S DECLARATION

I hereby declare that the project entitled **"To Develop an Online Production Scheduling Tool "**submitted in partial fulfillment for the award of the degree of Bachelor of Technology in 'Mechanical Engineering' completed under the supervision of **Dr. Bhupesh K Lad, Associate Professor Mechanical Engineering,** IIT Indore is an authentic work.

Further, I declare that I have not submitted this work for the award of any other degree elsewhere.

(A.Harshita)

Date:

CERTIFICATE by BTP Guide(s)

It is certified that the above statement made by the students is correct to the best of my knowledge.

Dr. Bhupesh K Lad

Associate Professor

Mechanical Engineering

Preface

This report on "The Development Of Online Production Scheduling Tool" is prepared under the guidance of Dr. B.K.Lad.

Through this report I have tried to give a detailed design of an online tool for Job shop Scheduling Problem and try to cover every aspect of the tool along with both the functions performed by the tool. I have tried to the best of my abilities and knowledge to explain the content lucidly. I have also added Tables and figures to make it more illustrative.

A.Harshita
B.Tech. IV Year
Discipline of Mechanical Engineering
IIT Indore

Acknowledgments

I wish to thank Dr. B.K.Lad for his kind support and valuable guidance.

I would like to thank Indian Institute of Technology Indore, India for providing experimental facilities and acknowledges financial support by the Project Number IAPP18-19/31 funded by the Royal Academy of Engineering London

A.Harshita
B.Tech. IV Year
Discipline of Mechanical Engineering
IIT Indore

Abstract

The project intends to develop an algorithm to solve a complex job shop scheduling problem (NP-Hard). The algorithm gives a near-optimal solution to the scheduling problem. We can get results for the problem with two different objectives one is minimum make-span and the second is the minimum number of late-jobs. This Schedule generation was mainly targeting the SMEs. Later we gave a scientific method to calculate the optimal due-dates for the industries who don't have due-dates or have wrong due-dates.

Table of Contents

List of Figures

List of Tables

Chapter 1:

Scheduling

1.1 Overview

The scheduling process comes into picture when resource availabilities are essentially fixed by the long term commitments of a prior planning decision. The four basic stages that describe the steps by which scheduling decisions are reached are formulation, analysis, synthesis, and evaluation.

- Formulation: Identifying a problem and deciding a criteria for decision
- Analysis: Identifying the decision variables, specifying the relations and also obeying the constraints among them. It is a detailed process of examining the elements of a problem and their interrelationships.
- Synthesis: To develop alternate solutions and also characterize the feasible options available.
- Evaluation: Comparing all feasible alternatives and selecting a desirable course of action.

1.2 Introduction

Scheduling is the task of determining when each operation is to start and finish. It is a tough job because each operation is in possible competition with the other ones for limited resources of time and capacity.

Scheduling is the allocation of resources over time to perform a collection of tasks. This rather general definition of the term does convey two different meanings. First scheduling is a decision-making function. Second scheduling is a body of theory as it is a collection of principles, models, techniques and logical conclusions that provide insight into the scheduling function.

The difference between "Scheduling" and "Sequencing" in a job shop process is that Scheduling is a process of assigning a start time and a completion time on a time scale of machine to each operation of each job keeping in mind the precedence constraint. However, a schedule cannot be determined with certainty because of variation in the arrival time of the job, operation processing time, as well as other attributes, are not fully known. On the other hand, sequencing means generation of flow order of jobs for each machine waiting in queue to be processed by a particular machine.

1.3 Motivation:

The problem that motivated this study is as follows: Suppose a scheduling problem consists of n jobs to be processed on m machines. Each job consists of a varied number of operations to be performed and all operations are to be performed based on the sequence in which they are presented to the machine. Each operation takes a particular amount of time to be processed on a particular machine. Every machine can process only one operation at a time. Taking a minimum number of late-jobs or minimum make-span as the deciding criteria the optimal schedule is to be generated.

Sequencing and scheduling problems occur in different industries and circumstances. The following are some examples of different situations which need sequencing or scheduling:

- parts waiting for processing in a manufacturing plant;
- aircraft waiting for landing clearance at anairport;
- computer programs running at a computing center;
- class scheduling in a school,
- patients waiting in a Doctor's office;
- ships to be anchored in a harbor, and
- Thursday afternoon chores at home.

1.4 Problem Definition:

The general definition of the sequencing problem can be stated as follows:
There are m machines $\{M_1, M_2, ..., M_m\}$ available and n jobs $\{J_1, J_2,..., J_n\}$ to be processed. A subset of these machines is required to complete the processing of each job. The flow pattern (process plan) for some or all jobs may or may not be fixed. Each job should be processed through the machines in a particular order that satisfies the job's technological constraints. The processing of job i on machine j is called an operation denoted by O_{ij}. Associated with each operation is a processing time denoted by P_{ij}, and a setup time denoted by S_{ij}. Also, associated with each job is a weight, w_i, a ready (release or arrival) time, r_i, and a due date, d_i. Finally, each job has an allowance time to be in the shop, a_i.

Thus, the general problem is to generate a sequence that satisfies the following conditions:

- all jobs are processed;
- all technological constraints are met for all jobs (feasibility condition), and
- all criteria that were selected areoptimized.

1.5 Assumption

A variety of assumptions are made in sequencing and scheduling problems. The assumptions vary with the problem. The following list contains typical assumptions generally applied to scheduling problem with variations depending on the situation.

- The set of the jobs and the set of the machines are known and fixed;
- All jobs and all machines are available at the same time and are independent;
- All jobs and machines remain available during an unlimited period;
- The processing time for each job on all machines is fixed, has a known probability distribution function, and sequence-independent;
- Setup times are included in processing times;
- A basic batch size is fixed for all jobs;
- All jobs and all machines are equally weighted;
- No preemption is allowed;
- A definite due date is assigned to each job;
- Each job is processed by all the machines assigned to it;
- Each machine processes all the jobs assigned to it,
- The process plan for each job is known and fixed.

1.6 Categories of Scheduling Problem:

A sequencing and scheduling problem is categorized as follows:

Deterministic sequencing and scheduling problems: when all elements of the problem do not include stochastic factors and are very well known in advance then it is said to be a deterministic sequencing and scheduling problem. These factors are the state of the arrival of the jobs to the shop, due- dates of jobs, ordering, processing times and availability of machines, etc.

Static sequencing and scheduling problems: the same as deterministic problems except that the nature of the job arrival is different. The set of jobs over time does not change, and it is available beforehand.

Dynamic sequencing and scheduling problems: the set of jobs and their arrival rates change over time.

Stochastic sequencing and scheduling problems: at least one of the problem elements includes a stochastic factor.

1.7 Methods Of Solution :

Several methods have been developed to solve and model sequencing and scheduling problems that belong to any of the four categories (deterministic, static, dynamic, and stochastic). These methods of solution can be classified as follows:

- Efficient optimal methods such as Johnson's algorithm to solve a flow shop problem with two machines and n jobs (Johnson 1954).
- Enumerative methods (implicit and explicit or complete) such as Brown and Lomnicki's branch and bound algorithm (Brown and Lomnicki 1966).
- Heuristic methods such as Campbell, Dudek, and Smith's algorithm to solve m machines and n jobs flow shop problems (Campbell, Dudek, and Smith 1970).
- Mathematical models (Integer Programming) such as Wagner's form to solve the permutation flow shop problem with n jobs and m machines (Wagner 1959).
- Heuristic search techniques: Simulated Annealing, Genetic Algorithms, Tabu Search, and Artificial Intelligence.
- Simulation models.
- Analytical models (such as Jackson's open queuing network model, Jackson 1957a).

Chapter 2

Job Shop Scheduling

2.1 Introduction

Job shop scheduling problem (**JSP**) is an optimization problem in which resources are assigned to jobs at particular times. We are given n jobs $J_1, J_2, \ldots, J_n$ of varying processing times, which need to be scheduled on m machines with varying processing power, while trying to minimize the makespan and late jobs.

The job shop consists of n jobs $J_1, J_2, \ldots, J_n$ where each job has varied number of operations $O_1, O_2, \ldots, O_n$ that need to be processed in a specific order (known as Precedence constraints). Each operation is processed on a specific machine and only one operation can be processed on a machine at a given time. A common relaxation of job shop is the **flexible** job shop where each operation can be processed on any machine of the shop floor i.e. all machines in the set are identical.

However, the possible number of schedules reaches to $(n!)^m$ for total enumerations.

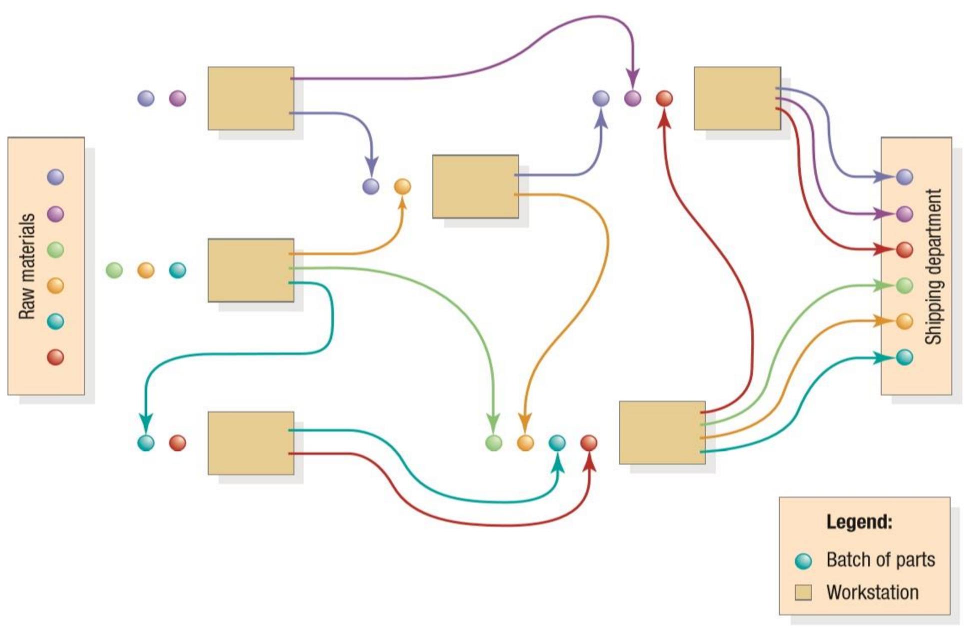

Fig 2.1: Job Shop Scheduling Representation

2.2 Flexible Job Shop Scheduling Problem:

The Flexible Job Shop Problem (FJSP) is an extension of the classical job shop scheduling problem which allows operation to be processed by any machine from a given set. The problem is to arrange the operations that are assigned to each machine such that the makespan of the entire schedule is minimized.

Flexible job shop closely imitates the actual production system because it breaks the restriction of unique allocation of each operation thereby allowing jobs to be processed by several different machines. The cycle time of each operation on the available machines is fixed and the setup times between the operations are either negligible or included in processing time. Each machine is continuously available from time zero and there are no precedence constraints among operations of different jobs, and each machine can process at most one operation at a time.

The FJSP can be solved by two approaches: hierarchical approaches and integrated approaches.

Hierarchical Approach: Hierarchical approaches considers assignment and sequencing independently i.e. assignment of operations to machines and the sequencing of operations on the machines are treated separately.

Integrated Approach: In integrated approaches, assignment and sequencing are not differentiated. Hierarchical approaches differ from the integrated approaches on the idea of decomposing the original problem to reduce its complexity by solving the routing and the scheduling as two subproblems. Whereas Integrated approaches considers both assignment and sequencing at the same time. These integrated approaches pave the way for formulating the multi-objective functions. In single-objective functions, makespan was only considered, whereas, in multi-objective function tardiness, due-date, critical machine work load, and total workload are considered. Multi-objective functions are formulated based on the requirement of the problem. Flexible job-shop problems can be differentiating in two kinds; i.e. total flexibility problem and partial flexibility problem.

1. Total flexibility: All the operations can be processed on all the available machines.

2. Partial flexibility: Only some operations can be performed on all machines but some operations are restricted to be performed on particular machines.

We are considering the partial flexibility in the current case. The schedule for the partial flexible Job Shop Scheduling problem is generated using a genetic algorithm.The way to solve the problem and produce the optimal solution is discussed in the next chapter.

2.3 Importance Of Job Shop Scheduling:

Having a schedule is essential for shop floor that is seeking to have visibility into manufacturing operations. This applies directly to job shop production in which the goal is to match overall supply and demand. Supply includes factors such as capacity, resource, labor & material availability in the job shop, while demand pertains to the actual job orders within the manufacturing facility according to the resource and capacities needed to adequately complete the job. Within job shop scheduling, the goal is to combine these components of internal supply & demand in the most efficient & optimal way possible.

Overall, utilizing the proper job shop schedule will allow you to properly handle the volatile job shop environment & ultimately boost production efficiency within your manufacturing operation.

2.4 Benefits of Job Shop Scheduling:

The benefits of job shop scheduling include the following:

- Visibility & Transparency: Visibility and transparency are important factors that affect a job shop schedule. An advantageous job shop scheduling should offer current order visibility, our labor, and machine capacity, waiting times, due-dates. Production gaining transparency over the production process is the overall goal of any scheduling tool. The schedule needs to provide all precise information needed to fully understand the components that influence the current & future manufacturing process. It is also beneficial to have a system that more people can access. This allows the general plan to be visible to more than one person in the work plant.
- Control and Precision over planning: On gaining visibility over the production process & understand the ongoing processes, you are then able to take over fullcontrol. With proper production scheduling, manufactures can anticipate & direct new orders as well as change current orders. You can move from being reactive to being proactive in shaping production.
- Identification of new opportunities: Identifying new opportunities can lead to substantial improvements within production. Similar to the way you identify bottlenecks through an increased level of visibility; new opportunities can popup as well. It may become apparent that a certain capacity is always idle & can be utilized elsewhere or a production line may be altered which would reduce its through-put time. Overall identifying opportunities are extremely important for supply chain growth.
- Real-time feedback: An advantageous job shop schedule will provide a manufacturer with notifications, such as error or other alerts about your production. These alerts allow you to make decisions based on real-time issues or hindrances within the production line such as when on-time delivery is in danger or when production is off schedule. This will aid in avoiding potential errors & increasing your productivity.
- What-if scenarios: What if scenarios can provide thorough visibility into what production schedules are the most beneficial to your manufacturing process. This feature allows you to simply add or change jobs & view the impact of small changes on other processes, resources, material, personnel & other – capabilities. This helps manufacturers locate quick solutions to various problems.

Chapter 3

Genetic Algorithm

Genetic Algorithm (GA) is an optimization technique inspired by Charles Darwin's theory of natural evolution. It reflects the process of natural selection where the fittest individuals are selected for reproduction to generate offspring of the next generation. Optimal solutions can be provided to difficult problems based on the principles of **Genetics and Natural Selection** which otherwise would take a lifetime to solve.

3.1 Introduction to Optimization:

Optimization is the process of making something better. In any process, we have a set of inputs and a set of outputs as shown in the following figure.

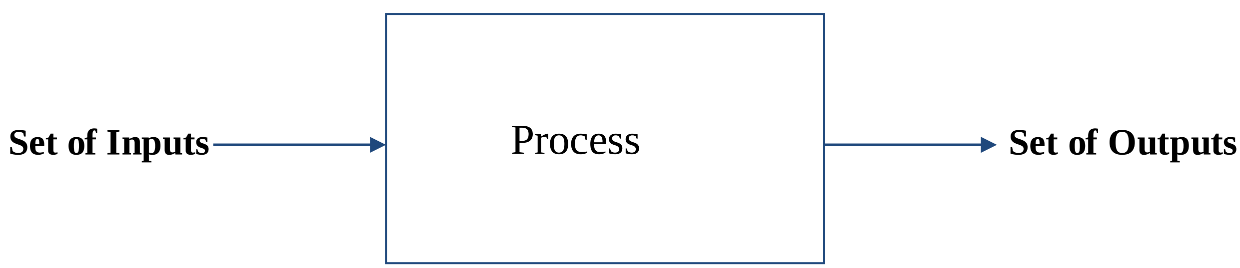

Fig 3.1 Optimization Block diagram

Optimization refers to giving out the best outputs for a given set of inputs. The definition of “best” is mutable for different problems, but in mathematical terms, it refers to minimizing or maximizing one or more objective functions, by varying the input parameters.

The inputs can be all possible solutions or values of the search space. Some of these points gives the optimal solution. In short, optimization is to find out that point or set of points from the search space

3.2 Introduction to Genetic Algorithms

Mankind has always derived its inspiration from nature. Genetic Algorithms (GAs) are search based algorithms based on the concepts of natural selection and genetics. GAs are a subset of a much larger branch of computation known as **Evolutionary Computation**.

Firstly fittest individuals from the population are selected then they undergo reproduction to produce offspring's which inherit characteristics of the parents & will be added to the next generation. If the parents have better fitness their offspring will be better than parents & have a better chance at surviving. The process keeps iterating until we find the desired fitness in the population.

In GAs, we have a pool of possible solutions to the given problem. These solutions then undergo crossover and mutation (like in natural genetics), producing new children, and the process is repeated over various generations. Each chromosome (or candidate solution) is assigned a fitness score (based on its objective function value) and the fitter individuals are given a higher chance to reproduce and yield more "fitter" individuals. This is in line with the Darwinian Theory of "Survival of the Fittest".

The initial population is iterated again and again to evolve better individuals or solutions over generations.

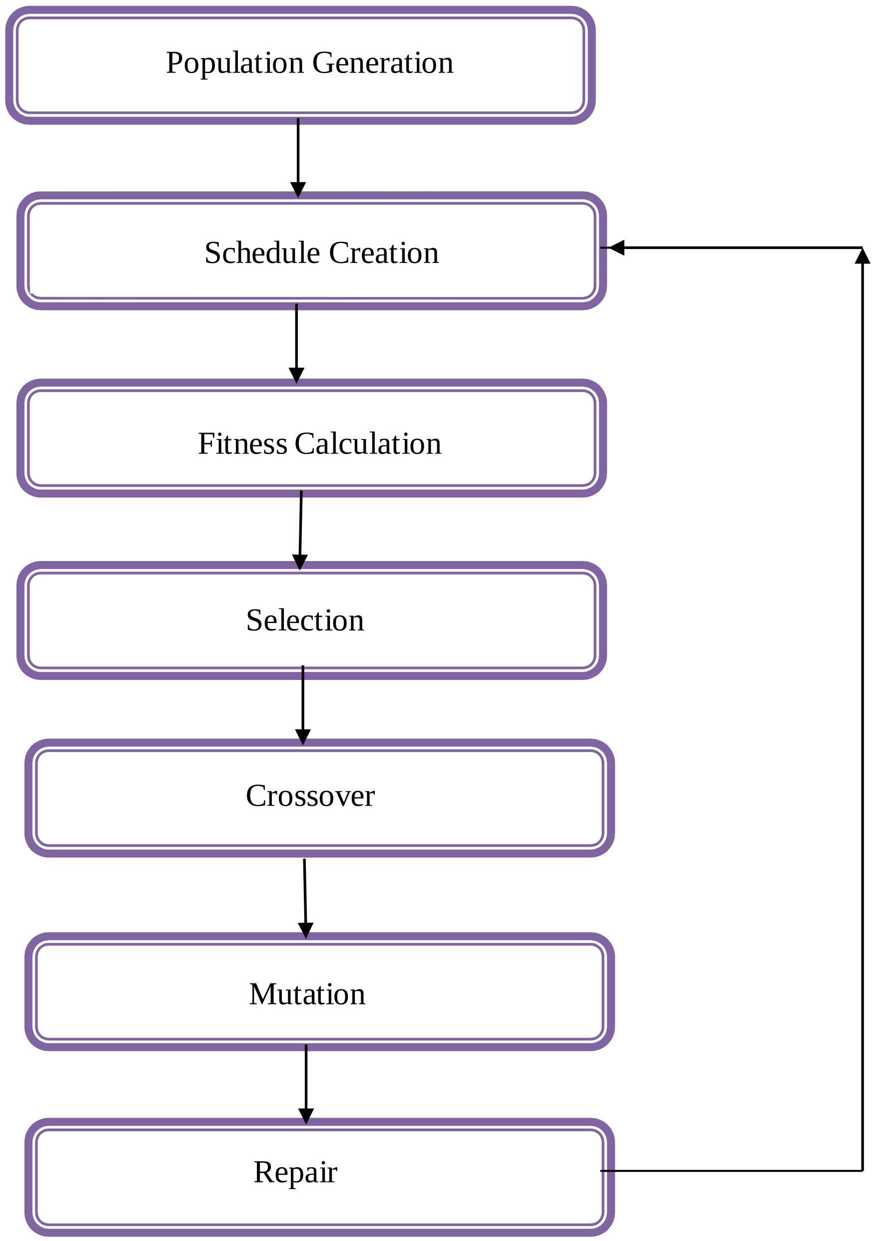

Fig 3.2 Genetic Algorithm Flowchart

3.3 Phases of Genetic algorithm

Population generation: Our first step is to generate a random population. Each individual of our Population will contain their own set of chromosomes.

Schedule Creation: The jobs waiting ahead of each machine are arranged in a proper sequence and hence a complete schedule for the job shop is generated.

Fitness Calculation: Here the fitness of each string is determined & a fitness score is given to each individual. The probability of an individual selected for reproduction is based on its fitness score.

Selection: Here the fittest individuals are selected and their genes are passed for the next generation. Two pairs of individuals are selected based on their fitness score. Individuals with the highest fitness score have more chance to be selected for reproduction.

Crossover: Crossover is the most significant phase in genetic algorithm. Based on the crossover probability we can predict how often crossover is performed. In the absence of crossover, offspring's will be the exact copy of parents. If crossover is done then the new individual is made up of parts of its parents. If crossover probability is 100% then all offspring are produced by crossover of the individuals. Crossover is made in the hope that new chromosomes will have good parts of old chromosomes & maybe the new chromosomes will be better. However, it is good to leave some parts of the population to survive to the next generation.

Mutation: In certain new offspring's are formed, some of their genes can be subjected to mutation with a low random probability. Some of the genes in the chromosome can be flipped. The main objective of mutation is to maintain diversity within the population and prevent premature convergence. Mutation probability gives the frequency of the parts of chromosomes undergoing mutation. Offspring is taken after crossover directly without any change in the absence of mutation. A part of the chromosome is changed on performing Mutation on it. If mutation probability is 0% nothing is changed if it is 100% whole chromosome is changed. Mutation is made to prevent falling GA into local extreme, but it should not occur very often because then GA will in fact change to random search.

Repair: The repair function corrects the invalid chromosomes generated after crossover and mutation taking care that the frequency of elements after repair is equal to the initial ones generated in the population generation function.

After repair function, the chromosomes are sent back again to the schedule creation function and the whole process is repeated in a loop until we reach the optimal solution.

3.4 Advantages And Limitations Of Genetic Algorithm

Advantages of GA:

- Easy to understand.
- Intrinsically parallel.
- Always answers.
- The answer gets better with time.
- Inherently parallel & easily distributed.
- Less time required for special application.
- Chances of getting optimal solutions are more.

Limitations of GA:

- The population size considered for evolution should be moderate for the problem (Ex: 20-30).
- Crossover probability should be in the range of 80-95%.
- Mutation probability should be in the range of 0.5-1% assumed as best.
- The method of selection should be appropriate.
- Writing of fitness function must be accurate.

Chapter-4

A Genetic Algorithm for Flexible Job Shop Scheduling Problem

4.1 Introduction

We have to recall that we can't solve the FJSP using the exact algorithm. Although exact methods based on disjunctive graph representation of the problem have been developed but they are limited to solve for a maximum of 20 jobs and 10 machines. Firstly to our knowledge, there are no approximate algorithms for producing solutions to FJSP with a guaranteed distance from the optimal solution.

In recent years several heuristic procedures such as dispatching rules, local search & meta-heuristic such as tabu search, simulated annealing & GAs have been successfully adopted to solve FJSP. They are all integrated approaches & differ from each other for initial population generation, different coding schemes, chromosome selection & offspring generation strategy. It consists of splitting the chromosome representation into 2 parts the first defining the routing policy & second the sequence of operation performed on each machine.

Jia et al proposed that distributed scheduling problem can be solved using modified GA & this method can be adapted for FJSP. For solving FJSP with recirculation Ho & Tay proposed an efficient methodology called GENACE based on a cultural evolutionary architecture. Finally, Kacem et al found a promising method for initial assignment by the use of chromosome representation that combines both routing & sequencing information & develops an approach by localization.

Dispatching rules are then applied to sequence the operation. Once the initial population is generated, use crossover & mutation operators to jointly modify assignment & sequences producing better individuals as the generations go by.

4.2 A GA for FJSP:

GA is a local search algorithm that follows the evolution paradigm. The algorithm applies genetic operators from the very starting i.e from the generation of the initial population. It aims at producing offsprings which are more fit than their ancestors. Every new individual (chromosome) at each generation corresponds to a solution i.e. a schedule of given FJSP instance. In the GA framework, more strategies can be adopted together to find individuals to add up the mating pool, both in the initial population phase and in the dynamic generation phase that's why GA is most powerful out of other local search algorithms. So a more variable search space can be explored at each algorithm step. The overall structure of our GA can be described as follows:

- Initial Population: The initial population is generated randomly by the coding scheme keeping in mind that although the sequence of operations of the job may vary but the number of operations to be performed on each job is fixed.

For example Size of initial population=4
Number of Jobs:4

Job	Operations
1	2
2	3
3	3
4	2

Table 4.1

The 4 different populations can be:

Number	Sequences
S1	1122233344
S2	1212323434
S3	1133322244
S4	1414232323

Table 4.2

As you can see it is just random shuffling of jobs keeping in mind that their operations frequency is fixed.

- Schedule Creation: The population sequence generated above is taken as the sequence for schedule creation considering each of the above chromosome as one schedule and the makespan is calculated for each sequence.

Job Type	Operation	Machine	Processing Time
1	1	1	18
1	2	3	14
2	1	2	13
2	2	1	15
2	3	2	10
3	1	3	8
3	2	3	6
3	3	1	5
4	1	1	12
4	2	2	15

Table 4.3

Gantt Charts for above 4 sequences:

S1:1122233344

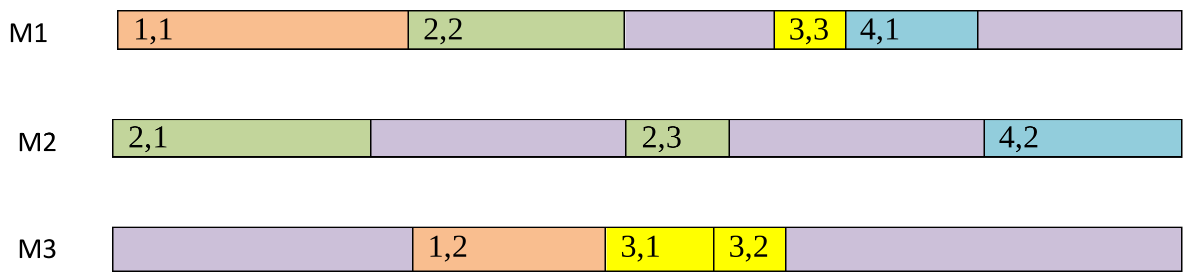

Fig 4.1

S2:1212323434

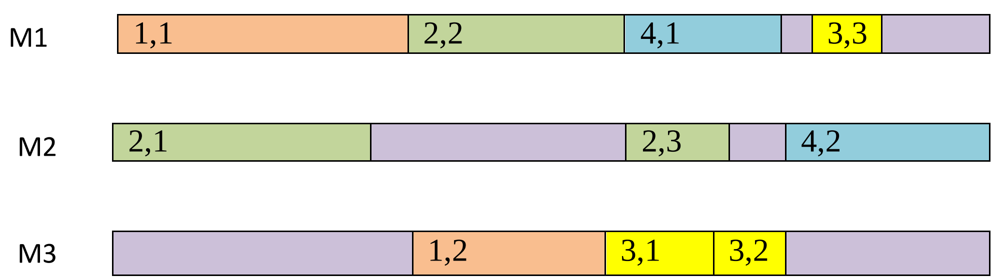

Fig 4.2

S3:1133322244

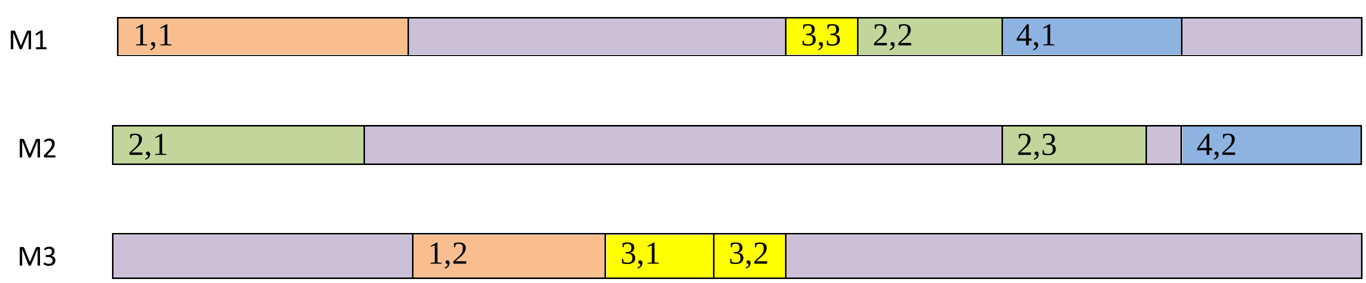

Fig 4.3

S4:1414232323

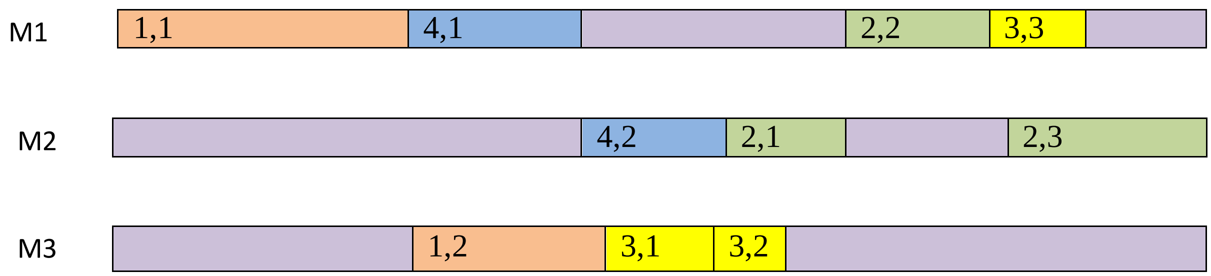

Fig 4.4

- Fitness Calculation: Based on the operation sequence a fitness score i.e makespan of each of the sequences is calculated. The makespan for the above 4 sequences are as follows:

Sequence	Makespan
1122233344	78
1212323434	60
1133322244	93
1414232323	73

Table 4.4

- Selection based on fitness score: From the fitness score we apply selection operator on each of the above sequences and hence get four best chromosomes out of any two chromosomes chosen randomly from the above four sequences.
- Crossover: After the selection of the chromosomes they undergo crossover based on crossover probability. The crossover probability is taken to be very high in the range of 80-90% the reason is as more and more chromosomes undergo crossover we will be getting better children out of the combinations of their parents.
- Mutation: After crossover, the chromosomes go for mutation. Generally, the mutation probability is very less but we have experimented with it by taking it a moderate value. The reason is we want to evolve some new solutions in the generation and better ones to be passed to the future.
- Repair: After crossover and mutation the frequency of operations on a particular chromosome might get changed hence the repair function checks for any abnormality present in the chromosome.

After one set of 4 sequences come out of the repair function they again go to the schedule creation then selection, crossover, mutation, and repair. This cycle goes on until we reach the optimal solution.

Chapter 5

Due Date Calculation

5.1 Introduction

There has been a long tradition of simulation research on scheduling of job shop. Much of the early work was concentrated on identifying rules that led to effective performance based on priority dispatching.

The objectives of scheduling are often multidimensional & there are many possible measures of scheduling performance. However, there are two primary factors: Shop time & due-date performance. The time that a job spends in the shop from the release of the order to completion is called its flow time. The mean job flow time is a basic measure of shop's performance at turning around orders, therefore it is often used as an indicator of success in responding quickly to customers. The average work in process level can be measured using mean flow time.

In actual job shops, meeting due-dates tends to be a more important criterion than minimal shop time. The study of due-date performance is also very much complicated as there is no universally accepted measure of effectiveness on this dimension. If a schedule meets all due-dates is a good one but when it is not possible to achieve this kind of perfection, how can the best level of performance be quantified? Along with the problem of measuring due-date performance, there is also the complication that no single priority rule dominates performance as compared to that of mean flowtime.

Researchers have advocated three main types of approaches in determining priorities using due-date information. These are:

- Allowance based priorities
- Slack based priorities
- Ratio based priorities

The time difference between the release date and due-date is called a job's flow allowance. Over time a job's remaining allowance shrink. Under allowance based priority rule the urgency of a job is related to its remaining allowance. If we are at time t, the remaining allowance of job j may be expressed as $a_j = d_j - t$, where $a_j(t)$ is the remaining allowance and $d_{j,}$ is the due date. A basic allowance –based priority system gives priority to the smallest $a_j(t)$ of course since t is the same for all jobs when we are making a dispatching decision, the job with the smallest $a_j(t)$ will also have the smallest d_j. Thus the smallest allowance- based rule is just the earliest due date (EDD) rule.

A job's slack time is its remaining allowance adjusted for remaining work. The slack for job j is $S_j = a_j(t) - p_{j,}$ where S_j is the slack time & $p_{j,}$ is the time required by the remaining operation of job j. The simplest slack based priority rule is the minimum slack time rule which gives priority to smallest s_j. In MST rules when two jobes have the same allowance the longer job is more urgent because its due-date allows less delay. SPT sequencing is considered the most efficient for meeting due-dates but MST incorporates some of the anti-SPT scheduling as compared to EDD priorities among the jobs with the same due-dates.

The third approach is similar to slack-based priorities but uses ratio arithmetic for implementation. For instance, the simplest form of the critical ratio is $a_j(t)/p_j$ or the remaining allowance divided by the remaining work. Here urgency is measured as the ratio of remaining allowance and remaining work.

The number of operations that are remaining to be performed on a job can be considered as one more factor of measuring of urgency. When two jobs have the same remaining allowance time and remaining work then the job with a larger number of remaining operations is considered as more important because it may encounter more number of chances of queuing, delay, etc. This reasoning has led to priority indices based on slack per operation (S/OPN) or remaining allowance per operation (A/OPN). Although these rules have performed well in some researches they along with SCR have some practical drawbacks. One problem is that the ratio may work in the wrong direction when their numerators are negative. When there are jobs with negative slack, the job with minimum slack-per-operation might not be the logical dispatching choice. Furthermore, the ratio priorities are dynamic: The relative priorities of two jobs may change as they wait in a queue. This feature could be complex in practice, although dynamic priorities are often considered to be desirable.

Operation milestone can be utilized to recognize the number of remaining operations. After a job’s due date is assigned it is possible to set a milestone in place to show when each operation should complete if the job is to process smoothly towards on-time completion. These milestones are called operational due-dates & the job's allowance is divided into as many pieces as the number of operations in the job. These pieces then play the role of operation flow allowances & they pace the job through the shop. Once operation due-dates have been established jobs can be dispatched by priority rules that utilizes only the operational due-dates & operational processing time in one of the three types of approaches:

The allowance based approaches thus leads to the earliest operational due-date (ODD) priorities;

The slack based approach leads to minimum operation slack time (OST);

The ratio-based approach leads to the smallest operation critical ratio (OCR).

The three measures of performances are: mean tardiness, proportion of jobs tardy & conditional mean tardiness.

5.2 Due date Tightness:

Absolute performance at meeting due-dates will be affected by how tight the due-dates are. For Example, Tighter due-dates tend to produce larger value of Mean Tardiness (MT) & Proportion of jobs tardy (PT) if other conditions are kept the same. Some evidences prove that due-date tightness does affect the relative performance of priority rules especially PT & MT.

5.3 Due date Rules:

A variety of decision rules can be used to set due-dates. If r_j denotes the time of arrival of job j, then we set the due date of job j as $d_j = r_j + a_j$ where $a_j = a_j(r_j)$ represents the original flow allowance. The following list describes many ways to set the original flow allowances (n_j denotes the number of operations for job j)

CON: $a_j = k$ (Constant flow allowance)

SLK: $a_j = p_j + k$ (equal slack)

NOP: $a_j = kn_j$ (proportional to number of operations)

PPW: $a_j = p_j + kn_j$ (Processing plus waiting time)

TWK: $a_j = kp_j$ (Proportional to total work)

The parameter k would be chosen differently for each rule in order to achieve a given average flow allowance. In addition, there is some evidence that the due-date rule is a significant factor.

Convay (1965b) tested NOP & CON with few priority rules & found that TWK gave better tardiness performance. However, the relative performance of two priority rules changed when the due-date rules were changed.

Kanet (1982) compared TWK, PPW, & NOP at various levels of due-date tightness & with several priority rules. It was noted that TWK was superior in terms of Mean tardiness (MT) performance. TWK is usually superior as well in terms of PT & MT.

Baker & Bertrand (1981, 1982) used a single machine model & compared TWK with CON & SLK. They produced results that showed the existence of a crossover effect in the choice of a due-date rule. In particular, TWK was the best rule tested except for very loose due-dates in which case SLK dominated.

5.4 Operational Milestone:

Not only are there alternative rules for setting job due-dates but there are also similar choices for setting operational due-dates. Once a job's due-dates are set, we divide its original flow allowance into as many segments as there are operations. These segments which determine operation due-dates can be constant for all operations of the given job. Alternatively, they can reflect equal stock or they can be proportional to total work. Then if we use the subscript (i,j) to denote the i^{th} operation of job j & adopt $d_{0j} = r_j$ we have

CON: $d_{ij} = d_{i-1j} + a_j/n_j$

SLK: $d_{ij} = d_{i-1j} + p_{ij} + (a_j - \Sigma p_{ij})/n_j$

TWK: $d_{ij} = d_{i-1j} + p_{ij} * a_j/\Sigma p_{ij}$

Kanet & Hayya (1982) compared CON & TWK as alternatives for setting due-dates are found TWK to be superior method. Using the TWK method, they found that operation based versions of EDD, MST & SCR produced better tardiness performance than the job-based versions.

5.5 Total Work Content Method

To summarize, we can say that the Total Work Content Method is one of the best methods to give optimal due-dates.

$$D_i = A_i + K \sum_{j=1}^{n_i} p_{ij}$$

D_i: Due-date k: due-date allowance factor

A_i: Arrival Date p_{ij}: Processing time of operation i of job j

Job Type	Operation	Machine	Processing Time	Number of jobs	Arrival Rate of Jobs	Service Rate of Machine	Total Flow Time
1	1	1	18	5	140	0.016	
1	2	3	14	5	140	0.0214	22
2	1	2	13	5	150	0.01578	
2	2	1	15	5	150	0.016	
2	3	2	10	5	150	0.01578	38
3	1	3	8	5	160	0.0214	
3	2	3	6	5	160	0.0214	
3	3	1	5	5	160	0.016	19
4	1	1	12	5	100	0.016	
4	2	2	15	5	100	0.01578	27

Table 5.1

Now calculate Arrival rate of Machines by:

Arrival Rate of Machine= Sum of Arrival rates of jobs on that machine

Machine	Arrival rate
1	140+150+160+100=550
2	150+150+100=400
3	140+160+160=460

Table 5.2

Machine	Arrival rate(per min) λ	Service rate (per minute) μ	Utilisation $=\lambda/\mu$
1	0.01273	0.016	0.7956
2	0.00926	0.01578	0.59
3	0.01065	0.0214	0.4976

Table 5.3

Job Id	Ready Time	Total Flow Time	Difference in days	Due-dates
1	27/11/19 9:30	22	0.015277778	27/11/19 9:52
2	27/11/19 9:30	38	0.026388889	27/11/19 10:08
3	27/11/19 9:30	19	0.013194444	27/11/19 9:49
4	27/11/19 9:30	7	0.004861111	27/11/19 9:37

Table 5.4

Slack=2 Formula: **Due-date = Ready time+ k*(Flow time)**

Job Id	Ready Time	(Total Flow time)*(Slack)	Difference in days	Due-dates
1	27/11/19 9:30	44	0.030555556	27/11/19 10:14
2	27/11/19 9:30	76	0.052777778	27/11/19 10:46
3	27/11/19 9:30	38	0.026388889	27/11/19 10:08
4	27/11/19 9:30	14	0.009722222	27/11/19 9:44

Table5.5

Chapter 6

Online Tool

6.1 Introduction

The online can be used to generate the optimal schedule for SMEs who don't have the costly software to generate one for them. Along with the schedule generation, it can be used to predict the optimal due-dates when industries don't have scientific methods to do so.

6.2 Importance:

The importance of optimal schedule generation is that it can properly handle a volatile job shop along with balancing of load on the machines. The importance of proper due-dates assignment is to ensure efficiency of industry because committing wrong due-dates to the customers can badly affect the reputation of the industry an furthermore cause disappointment to the customers.

6.3 Parts of Online Tool:

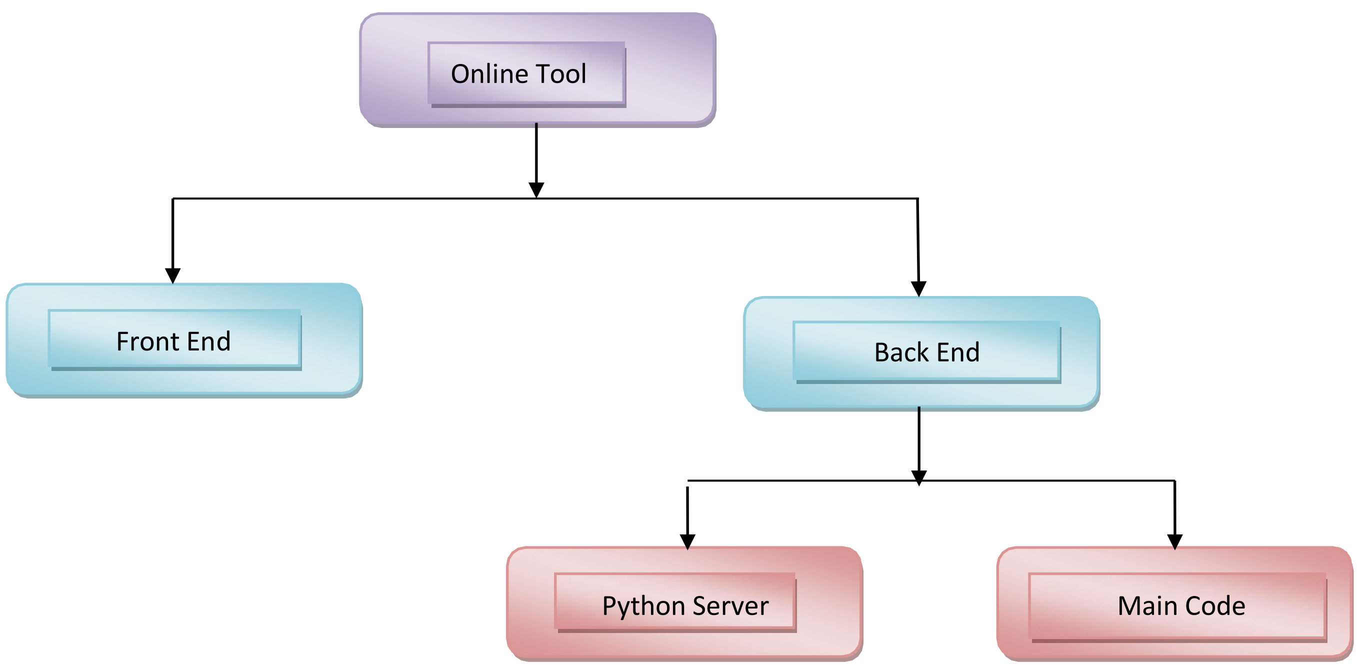

Fig 6.1 Flow chart of Online Tool

Front end: The HTML page is the front end of the tool which the user can see. After the file is uploaded the results are sent back to the user using the front end only.

Back End: The Back end of the tool comprises the processing and the calculations that take place in the background. There are two parts of the Back End.

Python Server: A python Server is nothing but a process that is running on your machine and does exactly two things :

- Listens for incoming requests on a specific TCP socket address (IP and a Port number)
- Handles the request and sends back a response to the user.

We have used a flask server to create a python server for the online tool.

Main Code: This code is used to evaluate the problem and return the answers,

The online tool looks like this:

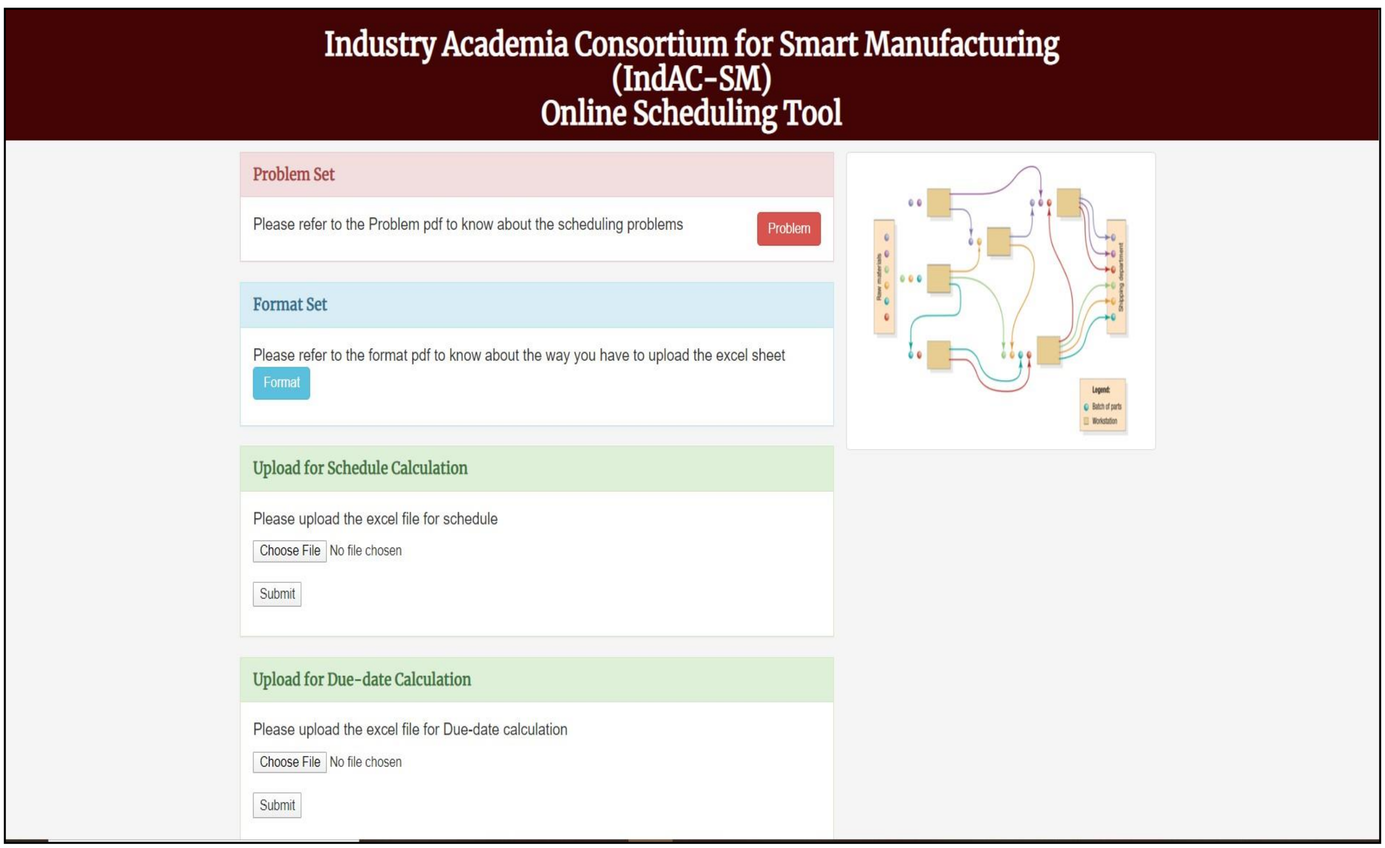

Fig 6.2 Online Scheduling Tool Frontpage

6.4 Functions performed by Online Tool:

As it is seen above the tool can be used to perform two different functions:

1. Schedule Generation: The tool can generate optimal schedules for job shop scheduling problem when part-id operation number machine on which the operation is to be performed and processing time are given as inputs. It gives back the make-span, number of late-jobs and the sequence of operations of jobs on each machine.
2. Due Date Calculation: The tool can give back the optimal due-dates to the user when the part-id, operation number, Machine on which the operation to be performed, Processing time of that operation, Arrival rate of jobs, Service rate of machines and ready time of jobs are given as input.

Chapter 7

Results and Discussion

7.1 Experimentation:

Both the functions of the tool are been subjected to rigorous testing and some of the results have been recorded by varying the various parameters.

For the schedule generation the parameters were:

- Number of iterations: As the number of iterations increases we approach towards a better solution, in other words, the solutions improve as time passes but time constraint is always there so we have to stop at a particular number of iterations so we have considered 50 iterations as optimal.
- Crossover probability: The value of crossover probability also affects the results as we go for higher values of crossover probability we get much better solutions. The reason is by crossover we have offspring have better qualities of their parents inherited in them. We have taken crossover probability as 90%.
- Mutation probability: The value of mutation probability has a huge impact on the results. The reason is going for a high value of mutation probability gives more randomness in the solution so the mutation probability is generally taken a very less value but we have experimented with this value by taking it a moderate one i.e. 15%.

Number of iterations	**Crossover Probability**	**Mutation Probability**	**Make-span**	**Late-jobs**
20	90	15	7753	2
50	90	15	6637	1
20	85	20	7598	3
50	85	20	7146	1

Table 7.1 Variation of Make-span and number of late-jobs with the number of iterations if crossover probability and mutation probability are kept constant.

Number of iterations: 20, Crossover probability: 90%, Mutation Probability: 15%

Make-span: 7753 Latejobs:2

Machine	Schedule
M1	[[10, 1], [5, 1]]
M2	[[10, 2], [9, 1], [10, 3], [7, 1]]
M3	[[6, 1], [6, 2], [9, 2], [9, 3], [3, 2], [2, 2], [1, 2], [7, 2], [7, 3], [2, 5], [7, 4], [7, 5], [8, 1], [10, 4], [8, 2], [10,5], [10, 6], [8, 3], [3, 5], [1, 6], [7, 11], [5, 4]]
M4	[[6, 3], [9, 4], [7, 6], [7, 7], [10, 7], [3, 6], [3, 7], [3, 9]]
M5	[[3, 3], [9, 5], [6, 4], [9, 6], [6, 5], [1, 3], [9, 7], [6, 6], [7, 8], [1, 7], [7, 9], [9, 13], [1, 8], [6, 10], [9, 14], [5, 3], [10, 11], [1, 10], [10, 12], [1, 11], [4, 8], [4, 9], [4, 11], [7, 16], [7, 20]]
M6	[[2, 3], [9, 8], [6, 7], [6, 8], [6, 9], [4, 5], [1, 9], [7, 12], [10, 10], [8, 4], [2, 9], [7, 17], [1, 13], [6, 13], [4,15]]
M7	[[1, 4], [4, 7], [4, 10], [5, 6], [10, 15], [2, 11], [10, 20]]
M8	[[3, 4], [2, 4], [9, 9], [9, 10], [9, 11], [1, 5], [4, 4], [4, 6], [5, 2], [10, 9], [5, 5], [2, 8], [7, 13], [3, 8], [7, 14], [8, 5], [1, 12], [7, 18], [8, 7], [9, 15], [4, 13], [10, 14], [8, 8], [9, 16], [2, 12]]
M9	[[2, 6], [10, 16], [3, 10]]
M10	[[9, 12], [7, 15], [4, 12], [6, 12], [7, 19], [5, 7], [4, 16]]
M11	[[4, 1], [4, 3], [2, 7], [8, 6], [3, 11], [7, 21]]
M12	[[7, 10], [6, 11], [1, 14], [4, 17]]
M13	[[1, 1], [2, 1], [3, 1], [4, 2], [10, 8], [10, 13], [4, 14], [10, 17], [8, 9], [2, 10], [6, 14], [10, 18], [10, 19], [2,13], [1, 15], [5, 8]]

Table 7.2: The sequences of operations when the number of iterations is 20, crossover probability= 90% and Mutation Probability= 15%.

Number of iterations:50, crossover probability:90%, mutation probability: 15%

Makespan:6637,   Latejobs:1

Machine	Schedule
M1	[[5, 1], [10, 1]]
M2	[[7, 1], [9, 1], [10, 2], [10, 3]]
M3	[[6, 1], [6, 2], [8, 1], [1, 2], [9, 2], [8, 2], [9, 3], [7, 2], [7, 3], [2, 2], [7, 4], [7, 5], [3, 2], [10, 4], [10, 5], [8, 3], [4, 6], [10, 6], [4, 7], [1, 6], [3, 6], [7, 16]]
M4	[[6, 3], [9, 4], [7, 6], [7, 7], [7, 14], [10, 14], [3, 9], [2, 9]]
M5	[[5, 2], [6, 4], [6, 5], [6, 6], [9, 5], [9, 6], [9, 7], [1, 3], [7, 8], [7, 9], [7, 10], [8, 4], [4, 9], [3, 4], [10, 9], [7, 15], [4, 11], [9, 12], [10, 11], [10, 13], [6, 13], [5, 8], [6, 14], [7, 18], [2, 12]]
M6	[[5, 3], [6, 7], [2, 3], [9, 8], [6, 8], [6, 9], [7, 11], [7, 12], [7, 13], [4, 12], [4, 13], [3, 5], [2, 5], [3, 7], [8, 8]]
M7	[[1, 4], [9, 11], [10, 12], [8, 7], [9, 14], [3, 11], [2, 13]]
M8	[[5, 4], [5, 5], [2, 4], [1, 5], [6, 10], [6, 11], [9, 9], [9, 10], [4, 8], [3, 3], [10, 8], [4, 10], [8, 5], [1, 7], [1, 8], [8, 6], [5, 7], [4, 14], [1, 10], [4, 15], [2, 8], [1, 12], [1, 13], [1, 14], [2, 11]]
M9	[[10, 7], [7, 17], [2, 6]]
M10	[[5, 6], [10, 10], [6, 12], [10, 16], [3, 8], [4, 16], [1, 15]]
M11	[[4, 1], [4, 3], [1, 9], [1, 11], [7, 19], [10, 19]]
M12	[[4, 4], [2, 7], [3, 10], [2, 10]]
M13	[[2, 1], [1, 1], [4, 2], [3, 1], [4, 5], [9, 13], [10, 15], [10, 17], [10, 18], [4, 17], [9, 15], [9, 16], [7, 20], [8, 9], [7, 21], [10, 20]]

Table 7.3: The sequences of operations when the number of iterations is 50, crossover probability= 90% and Mutation Probability= 15%.

Number of iterations:20 Crossover probability:85% Mutation Probability:20%

Make span : 7598 Latejobs:3

Machine	Schedule
M1	[[5, 1], [10, 1]]
M2	[[7, 1], [10, 2], [10, 3], [9, 1]]
M3	[[7, 2], [3, 2], [1, 2], [7, 3], [7, 4], [7, 5], [2, 2], [10, 4], [10, 5], [10, 6], [8, 1], [8, 2], [3, 5], [2, 5], [9, 2],[9, 3], [6, 1], [1, 6], [4, 4], [10, 14], [3, 7], [9, 12]]
M4	[[7, 6], [7, 7], [10, 7], [9, 4], [9, 5], [8, 3], [5, 7], [4, 7]]
M5	[[3, 3], [1, 3], [5, 2], [7, 8], [7, 9], [7, 10], [10, 8], [10, 9], [10, 10], [2, 6], [1, 8], [7, 12], [1, 9], [9, 6], [2,8], [8, 4], [3, 6], [6, 6], [4, 5], [9, 9], [1, 12], [2, 12], [1, 13], [1, 14], [4, 12]]
M6	[[5, 3], [2, 3], [7, 11], [10, 12], [10, 13], [9, 7], [7, 14], [10, 16], [7, 17], [5, 8], [10, 17], [4, 8], [3, 9], [8,7], [6, 12]]
M7	[[1, 4], [10, 11], [7, 15], [10, 18], [4, 9], [7, 18], [3, 11]]
M8	[[5, 4], [5, 5], [2, 4], [3, 4], [1, 5], [2, 7], [6, 2], [6, 4], [6, 5], [1, 10], [10, 15], [9, 8], [4, 6], [8, 6], [1, 11],[9, 11], [10, 19], [6, 9], [4, 10], [2, 13], [6, 11], [4, 13], [3, 10], [8, 8], [1, 15]]
M9	[[1, 7], [7, 16], [6, 7]]
M10	[[5, 6], [2, 9], [2, 11], [9, 10], [9, 13], [4, 11], [4, 14]]
M11	[[4, 1], [4, 3], [3, 8], [6, 8], [7, 19], [4, 15]]
M12	[[6, 3], [6, 10], [9, 14], [4, 16]]
M13	[[2, 1], [1, 1], [3, 1], [4, 2], [7, 13], [8, 5], [2, 10], [6, 13], [7, 20], [10, 20], [6, 14], [7, 21], [9, 15], [9,16], [8, 9], [4, 17]]

Table 7.4**:** The sequences of operations when the number of iterations is 20, crossover probability= 85% and Mutation Probability= 20%.

Number of iterations:50, crossover probability:85%, mutation probability: 20%

Makespan:7146, Latejobs:1

Machine	Schedule
M1	[[5, 1], [10, 1]]
M2	[[7, 1], [10, 2], [10, 3], [9, 1]]
M3	[[1, 2], [3, 2], [7, 2], [7, 3], [7, 4], [7, 5], [2, 2], [6, 1], [3, 5], [10, 4], [10, 5], [6, 2], [10, 6], [1, 6], [7, 11], [9, 3], [7, 12], [10, 10], [9, 4], [8, 2], [6, 5], [4, 9]]
M4	[[7, 6], [7, 7], [10, 7], [3, 7], [1, 8], [9, 5], [9, 6], [10, 15]]
M5	[[5, 2], [1, 3], [3, 3], [7, 8], [7, 9], [7, 10], [3, 6], [10, 8], [2, 5], [4, 6], [2, 6], [7, 13], [7, 14], [7, 15], [2, 7], [8, 3], [4, 10], [6, 7], [6, 8], [8, 4], [8, 5], [10, 16], [7, 20], [1, 12], [10, 17]]
M6	[[5, 3], [2, 3], [9, 2], [10, 9], [6, 3], [1, 9], [7, 17], [10, 12], [10, 13], [7, 19], [4, 12], [4, 13], [9, 9], [1, 11], [2, 11]]
M7	[[1, 4], [6, 4], [6, 6], [4, 14], [6, 10], [6, 12], [9, 16]]
M8	[[5, 4], [5, 5], [1, 5], [3, 4], [1, 7], [8, 1], [3, 8], [4, 7], [7, 16], [4, 8], [10, 11], [9, 7], [7, 18], [2, 8], [2, 9], [2, 10], [4, 15], [3, 10], [9, 11], [6, 9], [8, 8], [9, 12], [4, 16], [10, 19], [7, 21]]
M9	[[2, 4], [9, 8], [3, 9]]
M10	[[5, 6], [4, 11], [1, 10], [2, 12], [2, 13], [10, 18], [6, 11]]
M11	[[4, 1], [4, 3], [10, 14], [9, 10], [9, 14], [1, 13]]
M12	[[4, 4], [8, 6], [9, 13], [4, 17]]
M13	[[1, 1], [3, 1], [2, 1], [5, 7], [4, 2], [5, 8], [4, 5], [8, 7], [3, 11], [9, 15], [1, 14], [6, 13], [8, 9], [10, 20], [6, 14], [1, 15]]

Table 7.5**:** The sequences of operations when the number of iterations is 50, crossover probability= 85% and Mutation Probability= 20%.

Table 7.6: Comparing Results for different combinations of Crossover and Mutation Probability

Number of iterations	Crossover probability	Mutation Probability	Make-span	Late-Jobs	Average Make-span
50	90	5	7255 6937 7438	1 1 1	7210
50	90	10	7019 7122 7092	1 1 1	7077.67
50	90	15	7044 7062 6637	1 1 1	6914.33
50	85	10	7249 6926 7168	1 1 1	7114.33
50	85	15	7103 7077 7737	1 1 1	7155.33
50	85	20	6948 7034 7456	1 1 1	7146

So we choose 90% crossover probability as optimal

Table 7.7: Results when taking high values of Mutation probability

Crossover Probability	Mutation Probability	Make-span	Late-jobs
90	20	6889	1
90	20	7483	1
90	20	6955	1
90	20	6671	1
90	20	7061	1
90	20	7226	1

As it can be noted that ongoing for high mutation probability the range of values becomes very wide so we have taken 15% as ideal mutation probability.

7.2 Conclusion:

- The online tool can give optimal schedules to SME's which can't afford costly software.
- The tool can provide schedules for job shops with a large number of machines and jobs.
- The tool can be used to generate a schedule taking any of the objectives(minimum make-span or minimum late-jobs) as the main criteria to calculate the optimal schedule.
- The tool also gives the industries optimal due-dates which either don't know due-dates or have wrong due-dates.

References:

1. **Algorithms for Sequencing and Scheduling:** http://ikucukkoc.baun.edu.tr/lectures/EMM4129/Algorithms_for_Sequencing_and_Scheduling.pdf
2. **A Genetic Algorithm for Flexible Job Shop Scheduling Algorithm:** https://www.sciencedirect.com/science/article/abs/pii/S0305054807000524
3. **A Genetic Algorithm For Flexible Job Shop Scheduling:** https://www.researchgate.net/publication/239939866_A_Genetic_Algorithm_for_Flexible_Job_Shop_Scheduling
4. **Introduction to Genetic Algorithm:** https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3
5. **Parameters of GA:** https://www.obitko.com/tutorials/genetic-algorithms/parameters.php
6. **Genetic Algorithm:** https://www.geeksforgeeks.org/genetic-algorithms/
7. **Genetic Algorithm:** https://www.tutorialspoint.com/genetic_algorithms/genetic_algorithms_introduction.htm
8. **The Importance and Benefits of Job Shop Scheduling:** https://www.planettogether.com/blog/the-important-and-benefits-of-job-shop-scheduling
9. **Sequencing rules and Due-date assignments in job shop** https://pubsonline.informs.org/doi/10.1287/mnsc.30.9.1093
10. **Simulation modeling and analysis of due-date assignment methods and scheduling decision rules in a dynamic job shop scheduling production system:** https://www.sciencedirect.com/science/article/pii/S092552731000349X