CHAPTER 3

JOB SHOP SCHEDULING

This chapter introduces job shop scheduling problems with complete restrictions of basic problems. There are two ways of representing a job shop scheduling problem. One way is by disjunctive graph, which is a frame work to construct a schedule using the graph. The method of finding critical path in the graph is also presented. Another way is a Gantt chart representation to construct the schedule. Improvement of the constructed schedule using permutation method is discussed.

3.1 OVERVIEW

Basic JSSP is a static optimization problem, since all information about the production program is known in advance. General job shop problem is the probably most studied one by academic research during the last three decades and is notoriously difficult problem to solve. The JSSP is an NP (Nondeterministic Polynomial) hard problem and among those optimization problems, it is one of the least tractable known problem (Garey and Johnson 1979). It is purely deterministic, since processing time and constraints are fixed and no stochastic events occur. The JSSP also illustrates some of the demands required by a wide array of real world problems. In a shop floor, machines process jobs and each job contains a certain number of operations. Each operation has its own processing time and has to be processed on a dedicated machine. Each job has its own machine order and no relation exists between machine orders of any two jobs. For each job, the machine order of operations is prescribed and is known as technological production recipe or technological constraints, which are static to a problem instance.

Operations to be processed on one machine form an operation sequence for this machine. For a given problem, an operation sequence for each machine is called a schedule. Since each operation sequence can be permuted independently of operation sequences of other machines, the problem with n jobs and m machines can have a maximum of $(n!)^m$ different solutions The completion time of all jobs is known as makespan. The objective is to find a feasible schedule with minimum makespan. Feasible schedules are obtained by permuting the processing order of operations on machines without violating the technological constraints.

3.2 PROBLEM DEFINITION

Job shop scheduling problem consists of a set of jobs $J = \{1 \,..\, n\}$, a set of machines $M = \{1 \,..\, m\}$, where J_i denotes ith job ($1 \leq i \leq n$) and M_j denotes jth machine ($1 \leq j \leq m$). On the machines M_1, M_2, … M_m, the jobs J_1, J_2 … J_n are to be scheduled. Let V be the set of all operations in all jobs. Each job J_i has a set of operations o_{i1}, o_{i2}, …$o_{ik,}$ where k is total number of operations in the job J_i. Operation's precedence constraints are associated with each job and ensure that operation o_{ij} will be processed only after the processing of operation $o_{ij\text{-}1}$ in a particular job i.

Generally, standard model of n jobs, m machines job shop is denoted by $n/m/\varphi/P/C_{max}$. The parameter φ is technological matrix denoting the processing order of machines for different jobs. The machine order for ith job is given by φ_{ij} ($1 \leq j \leq m$), where j denotes jth operation in ith job. An example of the technological matrix φ can be represented as follows:

$$\varphi = \begin{pmatrix} M_2 & M_3 & M_1 \\ M_1 & M_2 & M_3 \\ M_3 & M_1 & M_2 \end{pmatrix}$$

Each row of the above matrix represents a job. For first job, first operation is performed on machine M_2, second operation is performed on machine M_3 and third operation is performed on machine $M_{1.}$ Similarly, other jobs are executed on different machines.

Matrix P, denoting the processing time of different operations, is represented as follows:

$$P = \begin{pmatrix} p_{11} & p_{12} & p_{13} \\ p_{21} & p_{22} & p_{23} \\ p_{31} & p_{32} & p_{33} \end{pmatrix}$$

where p_{ij} represents time of jth operation of ith job.

The technological matrix φ and the processing time matrix P are given as problem data. The processing order (machine sequence) for machine M_i is given by Π_{ik} $(1 \le k \le n)$, where k denotes kth operation to be processed on machine M_i. A solution to JSSP can be represented by a matrix Π denoting processing orders of all machines. For instance, one solution of the above problem is considered as follows:

$$\Pi = \begin{array}{c} \\ M_1 \\ M_2 \\ M_3 \end{array} \begin{array}{c} \begin{array}{ccc} o_1 & o_2 & o_3 \end{array} \\ \begin{pmatrix} J_2 & J_3 & J_1 \\ J_1 & J_2 & J_3 \\ J_3 & J_1 & J_2 \end{pmatrix} \end{array}$$

According to the above schedule, first operation of second job is scheduled on machine M_1, followed by second operation of third job and third operation of first job. Similarly other machines have schedules represented in second and third rows. Generally, subscript values denoting machine numbers in φ and job numbers in Π are given to formulate technological matrix and matrix representing a solution respectively. Processing unit of j^{th} operation of i^{th} job on a machine is denoted as o_{ij}. Each operation o has at most two direct predecessor operations, a job predecessor PJ_o and a machine predecessor PM_o. First operation of a machine sequence has no PM_o, and first operation of a job has no PJ_o. Similarly each operation has at most two direct successor operations, a job successor SJ_o and a machine successor SM_o. Last operation of a machine sequence has no SM_o and last operation of a job has no SJ_o. An operation o is called schedulable, if both, PJ_o and PM_o are already scheduled.

Let $r_{o_{ij}}$ be the starting time of j^{th} operation of i^{th} job. Completion time $C_{o_{ij}}$ for o_{ij} is calculated as in Equation (3.1)

$$C_{o_{ij}} = r_{o_{ij}} + P_{ij}, \qquad r_{o_{ij}} = \max(C_{PJ_{o_{ij}}}, C_{PM_{o_{ij}}}) \qquad (3.1)$$

$r_{o_{ij}}$ are assigned by zero values for undefined $PJ_{o_{ij}}$ and $PM_{o_{ij}}$. After scheduling all operations, the makespan C_{max} representing completion time of all operations is calculated as in Equation (3.2).

$$C_{max} = \max(C_{o_{ij}}) \quad \text{for all } o_{ij} \in V \qquad (3.2)$$

Main objective is to minimize C_{max} value with certain restrictions listed as follows:

- No two operations of one job may be processed simultaneously.
- A machine performs only one job at a time.
- Once an operation is initiated for processing, it will not be interrupted until its completion.
- An operation of a job can not be started until its previous operations of the same job are completed
- More than one operations of a job cannot be processed on a single machine.
- Jobs must wait for the next machine to be available.
- Machines may be idle within the schedule period.
- One job is independent with other jobs.

In real world problems, the set of constraints is much more complex. Only a few assumptions of the basic JSSP may hold in practice. Hence, JSSP is popular in academic research as a test-bed for different solution techniques to solve combinatorial optimization problems.

3.3 DISJUNCTIVE GRAPH MODEL

It is useful to define a problem into a graph (Dorigo et al 1996) for JSSP. For an instance of a JSSP, there can be a disjunctive graph $Ģ = (V, A, E)$, where V is a set of nodes corresponding to different operations in the job shop, A is a set of conjunctive arcs representing precedence of operations in jobs and E is a set of disjunctive arcs with no direction representing machine processing constraints. The set V also includes two dummy operations representing a start operation and an end operation of a schedule.

As an example, JSSP with four jobs and three machines may now be considered. The data are given in Table 3.1.

Table 3.1 Processing Time and Operation Sequence for a 4 × 3 Instance

Job	Processing Sequence
J_1	(1,2) (2,3) (3,4)
J_2	(3,4) (2,4) (1,1)
J_3	(2,2) (3,2) (1,3)
J_4	(1,3) (3,3) (2,1)

According to Table 3.1, job shop scheduling problem has four jobs, three machines and twelve operations. Processing sequence for each job is a set of items (*m*, *t*), where *m* denotes machine number and *t* denotes time of execution of this operation on the machine *m*. According to this problem, technological matrix φ and operation time matrix P are given below:

$$\varphi = \begin{pmatrix} M_1 & M_2 & M_3 \\ M_3 & M_2 & M_1 \\ M_2 & M_3 & M_1 \\ M_1 & M_3 & M_2 \end{pmatrix} \qquad P = \begin{pmatrix} 2 & 3 & 4 \\ 4 & 4 & 1 \\ 2 & 2 & 3 \\ 3 & 3 & 1 \end{pmatrix}$$

One solution to this problem Π can be represented as follows:

$$\Pi = \begin{pmatrix} J_1 & J_4 & J_3 & J_2 \\ J_1 & J_3 & J_2 & J_4 \\ J_2 & J_3 & J_1 & J_4 \end{pmatrix}$$

Disjunctive graph of the above problem is shown in Figure 3.1. In the graph, vertices drawn as circles represent operations. Arcs in A and E are weighted with the processing time of operation representing the source node v of arc (v, w). Hence, arcs starting at operation v are identically weighted. Arc weights stand for the processing time obtained from the matrix P. According to φ, related machine for first operation of first job is M_1. The matrix element p_{11} contains its processing time as 2. Thus directed arcs, which have first operation of first job as their node, are weighted with a processing time of 2 units. Within the set of conjunctive arcs A, dummy operation s is connected to first operation of each job and arcs from s are weighted with zero. Last operation of each job is incident to end operation e and consequently weighted with the processing time of last operation in each job. The end operation e is also a dummy operation.

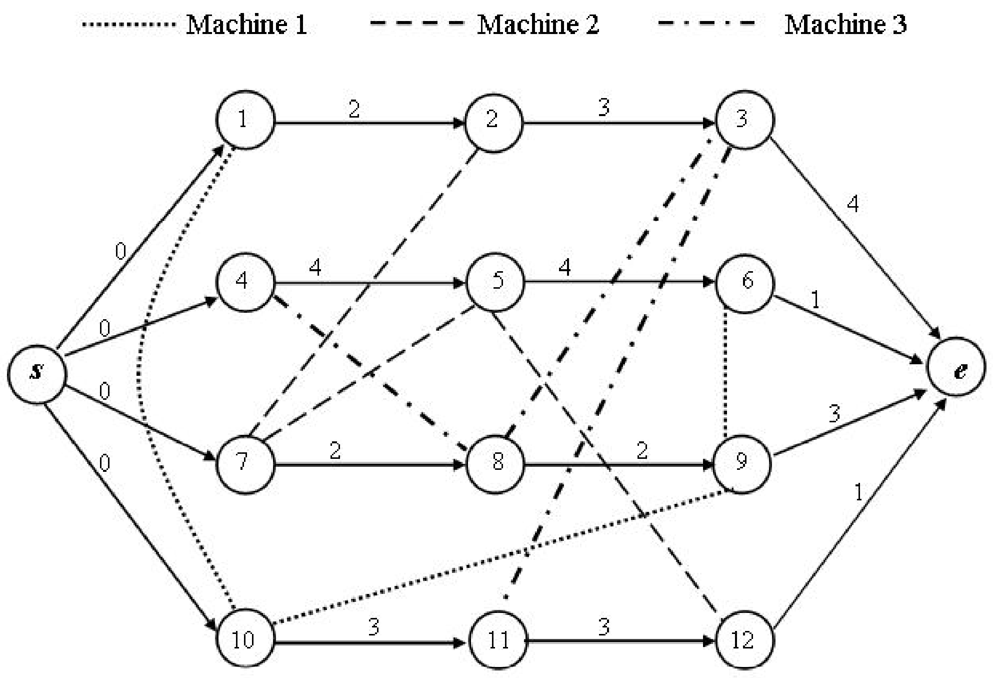

Figure 3.1 Disjunctive Graph Representation for 4 × 3 Problem in Table 3.1

Directed arcs in the set A represent processing order of operations in a single job. For instance operations 1, 2 and 3 belong to first job and have to be processed in the order given by the directed arcs (1,2) and (2,3). Furthermore, arcs (s, 1) and (3, e) connect first and last operation of the first job, where s and e are dummy operations denoting begin and end of the entire schedule. Undirected arcs in the set E represent machine processing constraints, which are obtained from the matrix φ. For example, operation 1 is first operation of first job, operations 6 and 9 are in third position of second and third jobs respectively and operation 10 is first operation of fourth job. These four operations have to be processed on M_1 as given by φ_{11}, φ_{23}, φ_{33} and φ_{41}. In this example, one subset in E (say E_1) consists of undirected arcs, which fully connect operations 1, 6, 9 and 10. Theoretically, each of four operations can precede all other operations of M_1 such that arbitrary machine sequence Π_1 can be obtained for M_1. Likewise, remaining undirected arcs are used to obtain machine sequences Π_2 and Π_3 for second and third machines respectively.

3.4 SECULAR SCHEDULING

This section presents a simple framework for building a schedule from the scratch by using a graph independent of assigned machine for all operations. A graph with only conjunctive arcs is shown in Figure 3.2. Initially, a set {1, 4, 7, 10} consisting of first operation of each job in the problem instance is created. An operation v is called schedulable in the next stage, if its predecessors PM_v and PJ_v are already scheduled. Number of operations of the problem instance determines number of stages in the scheduling process. Let $\Re \subset V$ be a set of all schedulable operations at stage t and $K \subset V$ be a set of operations scheduled during the scheduling process. Initially, $\Re$ contains first operation of each job, that is, successors of the start operation *s*. K is empty, because no operations have been scheduled at first stage.

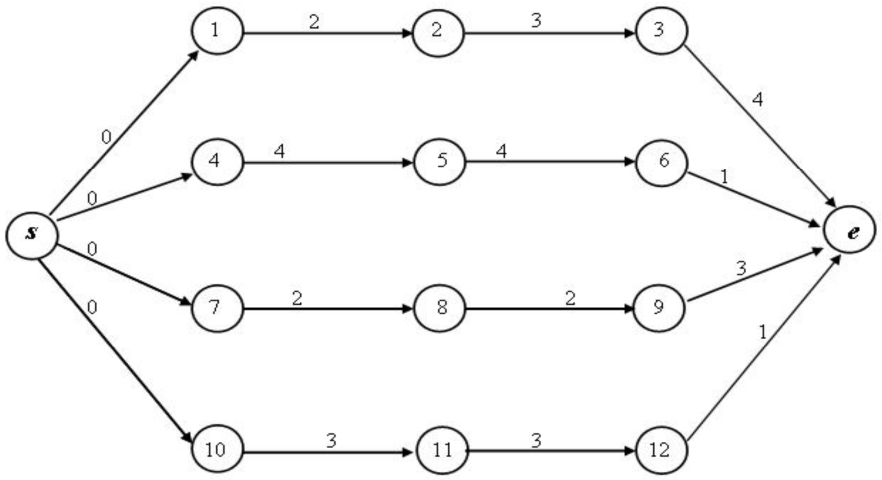

Figure 3.2 Graph Representation with Job Constraints

In order to determine an operation v to be scheduled next, a choice operator Φ, is declared. Once an operation v is chosen, v is deleted from $\Re$ and is also added to K**.** Job successor SJ_v of v is added to $\Re$ after scheduling v. Hence, scheduling an operation v at stage t means to add the operation v to the set K. This is done by constructing machine constrained arc (w, v) such that the operation $w \in K$ (w has been scheduled last on its machine) and the arc $(w, v) \in E_i$ (w and v are to be processed on the same machine M_i). If first operation is scheduled on machine M_i, no arc is constructed. Each time an operation has been scheduled, w is replaced by v in K. The operator Φ determines a control strategy of the scheduling algorithm and hence, this operator Φ should be effectively modeled to produce better solution. The working of this method is described below to find an example for a schedule.

As noted earlier in this section, the set $\Re$ contains {1, 4, 7, 10}. Suppose the operator Φ selects an operation 1 among operations in the set $\Re$.

Now this operation 1 is added to the set K and is also deleted from the set $\Re$. Successor of operation 1 is operation 2, which is added to the set $\Re$. Now sets $\Re$ and K have following elements.

$\Re = \{2, 4, 7, 10\}$ and $K = \{1\}$

In second stage, the operator Φ may select an operation 4. This operation 4 is added to K and is also removed from $\Re$. The successor of 4 is 5, which is added to $\Re$. Now sets $\Re$ and K contain following elements.

$\Re = \{2, 5, 7, 10\}$ and $K = \{1, 4\}$

In third stage, the operator Φ may select an operation 2. This operation 2 is added to K and is removed from $\Re$. Successor of 2 is 3, which is added to $\Re$. Now sets $\Re$ and K contain following elements.

$\Re = \{3, 5, 7, 10\}$ and $K = \{1, 4, 2\}$

In fourth stage, the operator Φ may select an operation 7. This operation 7 is added to K and is removed from $\Re$. Successor of 7 is 8, which is added to $\Re$. Sets $\Re$ and K contain following elements.

$\Re = \{3, 5, 8, 10\}$ and $K = \{1, 4, 2, 7\}$

In fifth stage, the operator Φ may select an operation 10. This operation 10 is added to K and is removed from $\Re$. Successor of 10 is 11, which is added to $\Re$. Sets $\Re$ and K contain following elements.

$\Re$ = {3, 5, 8, 11} and K = {1, 4, 2, 7, 10}

In sixth stage, the operator Φ may select an operation 5. This operation 5 is added to K and is removed from $\Re$. Successor of 5 is 6, which is added to $\Re$. Sets $\Re$ and K contain following elements.

$\Re$ = {3, 6 , 8, 11} and K = {1, 4, 2, 7, 10, 5}

In seventh stage, the operator Φ may select an operation 8. This operation 8 is added to K and is removed from $\Re$. Successor of 8 is 9, which is added to $\Re$. Sets $\Re$ and K contain following elements.

$\Re$ = {3, 6, 9, 11} and K = {1, 4, 2, 7, 10, 5, 8}

In eighth stage, the operator Φ may select an operation 3. This operation 3 is added to K and is removed from $\Re$. Successor of 3 is *e*, which cannot be added to $\Re$. Sets $\Re$ and K contain following elements.

$\Re$ = {6, 9, 11} and K = {1, 4, 2, 7, 10, 5, 8, 3}

In ninth stage, the operator Φ may select an operation 11. This operation 11 is added to K and is removed from $\Re$. Successor of 11 is 12, which is added to $\Re$. Now sets $\Re$ and K contain following elements.

$\Re$ = {6, 9, 12} and K = {1, 4, 2, 7, 10, 5, 8, 3, 11}

In tenth stage, the operator Φ may select an operation 9. This operation 9 is added to K and is removed from $\Re$. Successor of 9 is *e*, which cannot be added to $\Re$. Sets $\Re$ and K contain following elements.

$\Re = \{6, 12\}$ and $K = \{1, 4, 2, 7, 10, 5, 8, 3, 11, 9\}$

In eleventh stage, the operator Φ may select an operation 6. This operation 6 is added to K and is removed from $\Re$. Successor of 6 is e, which cannot be added to $\Re$. Sets $\Re$ and K contain following elements.

$\Re = \{12\}$ and $K = \{1, 4, 2, 7, 10, 5, 8, 3, 11, 9, 6\}$

In twelfth stage, the operator Φ may select an operation 12. This operation 12 is added to K and is removed from $\Re$. Successor of 12 is e, which cannot be added to $\Re$. Sets $\Re$ and K contain following elements.

$\Re = \{\ \}$ and $K = \{1, 4, 2, 7, 10, 5, 8, 3, 11, 9, 6, 12\}$

This procedure stops, if the set $\Re$ becomes empty and the set K has a schedule of all operations. During the construction of solution, an operation can be selected from $\Re$ such that the makespan of the partial solution built so far is least worsened. Once the operation is selected for a schedule, this selected operation remains fixed up to the end of the insertion procedure. In early stages, these algorithms perform excellent. But in later stages, they suffer from a shrinking of the set $\Re$.

3.5 SCHEDULE REPRESENTATION

Set K represents a feasible schedule, in which the position of an operation gives the precedence of that operation. That is, operation $K_i \in K$ ($i \leq 2 \leq$ total operations) is logically processed after the processing of operation $K_{i\text{-}1}$. Suppose operations processed on first machine, second machine and third machine are ordered by {1, 10, 9, 6}, {2, 7, 5 12} and {4, 8, 3, 11} respectively. Hence, the graph *Ģ* has three sets of disjunctive arcs {(1, 10), (10, 9), (9, 6)}, {(2, 7), (7, 5), (5, 12)} and {(4, 8), (8, 3), (3,11)} corresponding to first machine, second machine and third machine respectively. Hence, undirected arcs given in Figure 3.1 are replaced by directed arcs according to the machine processing order denoted by the respective set for a machine. Figure 3.3 represents the constructed schedule or the current solution *K* from the previous section.

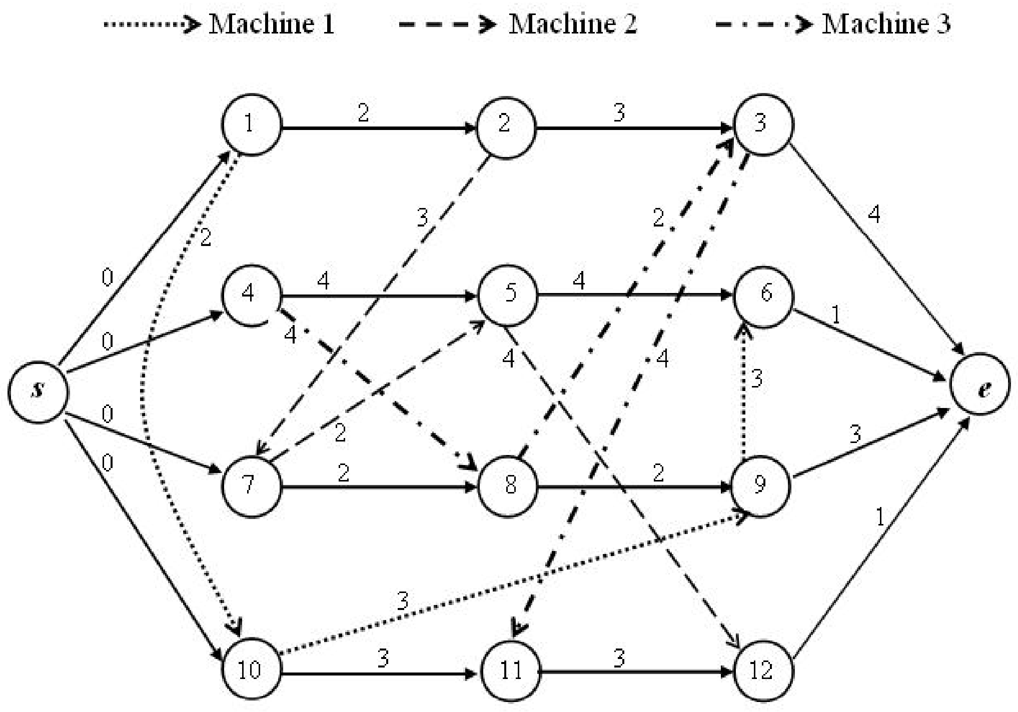

Figure 3.3 Graph Representation for a Constructed Schedule

According to the solution represented by the above graph, each machine has following sequence of operations belonging to different jobs.

Sequence of operations in machine one :

$$\begin{tikzcd} 1 \arrow[r] & 10 \arrow[r] & 9 \arrow[r] & 6 \end{tikzcd}$$

Sequence of operations in machine two :

$$\begin{tikzcd} 2 \arrow[r] & 7 \arrow[r] & 5 \arrow[r] & 12 \end{tikzcd}$$

Sequence of operations in machine three :

$$\begin{tikzcd} 4 \arrow[r] & 8 \arrow[r] & 3 \arrow[r] & 11 \end{tikzcd}$$

3.6 CRITICAL PATH

Makespan of a schedule is equal to length of the longest path in *Ģ*. Thus solving a JSSP is equivalent to finding a complete schedule that minimizes the length of the longest path in the directed graph *Ģ*. Many changes to the schedule do not affect the makespan value. Methods used for solving scheduling problems try to find out operations, which influence the makespan value and for this reason, the term critical path has been invented (Blazewicz et al 1993). This concept appears mostly in relation to job shop scheduling, but it can be used for other kinds of scheduling as well. Critical path should consist of operations whose delay would increase the makespan. There can be more than one critical path in a problem instance. Most of the algorithms employ critical paths to find schedules with minimum makespan values. A move is performed by the reversal of a single directed disjunctive arc on a critical path (Barnes and Chambers 1995). Informally the definition says that operation (j, m) is critical, if the makespan depends on the length of (j, m).

Critical path can be found by using two kinds of values, namely forward and backward values, associated with each node of the graph *Ģ*. Forward and backward values of a node are the latest starting time and the earliest finishing time of an operation associated with the node. Let $\mu(k)$ be the operation time for k^{th} node in a single dimension. Forward value f_T of an operation T is calculated as in Equation (3.3).

$$f_T = \max(r_{PJ_T} + \mu(PJ_T), \quad r_{PM_T} + \mu(PM_T)) \tag{3.3}$$

Backward value b_T of an operation T is calculated as in Equation (3.4).

$$b_T = \min(f_{SJ_T} - \mu(SJ_T), \ f_{SM_T} - \mu(SM_T)) \tag{3.4}$$

Each node in the graph is labelled by a pair (f_T, b_T). If f_T is equal to b_T for any node i ($1 \leq i \leq$ total nodes), this node i is on the critical path. Figure 3.4 shows a critical path of a schedule represented in Figure 3.3.

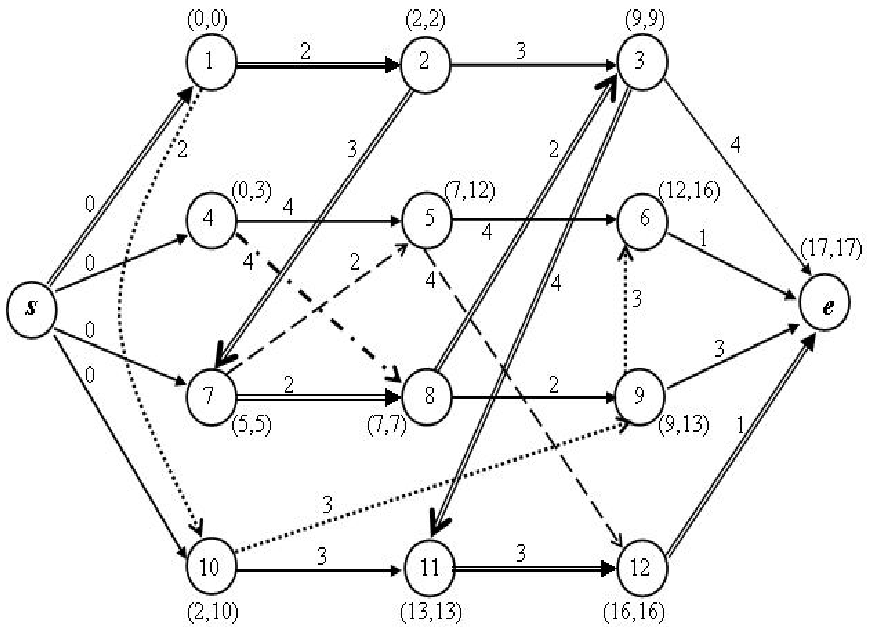

Figure 3.4 Graph with the Latest Starting Time, the Earliest Finishing Time and Critical Path

The critical path produced from Figure 3.4 is denoted as follows:

$$\begin{tikzcd} \boxed{s} \arrow[r] & \boxed{1} \arrow[r] & \boxed{2} \arrow[r] & \boxed{7} \arrow[r] & \boxed{8} \arrow[r] & \boxed{3} \arrow[r] & \boxed{11} \arrow[r] & \boxed{12} \arrow[r] & \boxed{e} \end{tikzcd}$$

A critical block, which is a maximal sequence of consecutive critical operations on the same machine, is found and elements of a critical block can be exchanged. However, consecutive critical operations in the same job cannot be exchanged, because it would violate the job shop constraints.

3.7 GANTT CHART

An intuitive way of representing a JSSP schedule is a Gantt chart. Figure 3.5 shows a Gantt chart for the schedule represented in Figure 3.3, which corresponds to the problem in Table 3.1. Gantt chart represents position of operations on the respective machines with idle time. In Gantt chart, operation blocks are moved to the left as much as possible to have a compact schedule, which has the best makespan value.

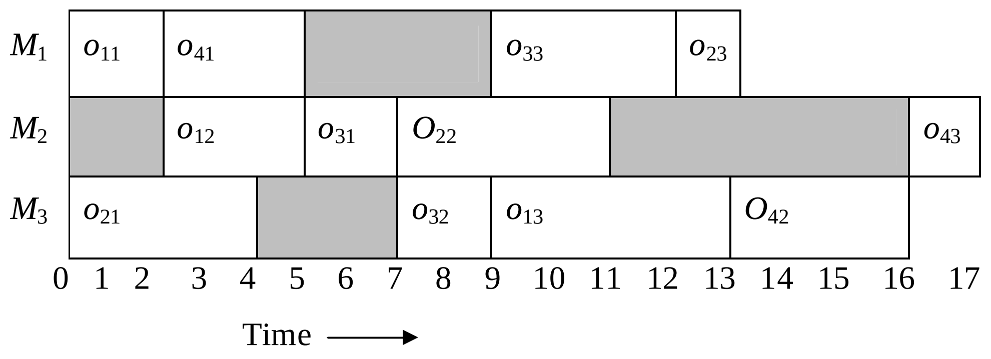

Figure 3.5 Feasible Gantt Chart for 4 × 3 Problem with Blocks

Machine M_1 processes first operation of first job, first operation of fourth job, third operation of third job during the time intervals between 0 and 2, 2 and 5, and 9 and 12 respectively and third operation of second job during 13th unit time. Idle time is marked by shaded rectangle. Likewise, other operations are processed in machines M_2 and M_3. Construction of this Gantt chart is described as follows:

Time units and machine numbers are represented by x axis and y axis respectively. Each row i consists of operations to be processed on M_i in the order given by Π_i. Sequence of machine M_1 denotes the processing of first operation of first job followed by first operation of fourth job, third operation of third job and third operation of second job. Operations are depicted in the length of their processing time. Third operation to be processed on machine M_1 is considered for example. Its job number J_3 is obtained from Π_{13} and then its processing time of this operation o_{33} is 3, which is obtained from P_{33}. Position of operations in the technological order of their job is determined. For instance, second job to be processed on M_1 is J_4. Now φ_4 is scanned for M_1 and φ_{41} with $j = 1$ is found. Thus, the operation considered is o_{41}, which is to be processed as first operation of J_4. Operations in the Gantt chart are rearranged in such a way that an operation with a lower index j of some job precedes an operation with a higher index j. Now o_{42} is to be processed as second operation of J_4. Hence, operation o_{41} has to precede o_{42} in order to avoid a simultaneous processing of operations of one job. Starting time and completion time of operations can now be taken directly from x axis of the Gantt chart. Dependencies of job and machine predecessors are outlined explicitly by machine idle times (in gray shade). Completion time of the rightmost operation in the Gantt chart gives the makespan achieved. In the example $C_{o_{43}}$ defines the C_{max} value and has the value as 17.

3.8 SEMI-ACTIVE AND ACTIVE SCHEDULE

A schedule, in which an operation starts after its earliest possible starting times, is known as passive scheduling. The earliest possible starting time can be defined as the actual starting time of operations. A schedule, in which an operation starts at its earliest starting time, is known as semi-active scheduling. Semi-active schedule cannot be improved in terms of makespan without changing operation sequence of machines. A schedule, in which improvement of makespan cannot be gained even by changing any of the processing orders Π_i, is called active schedule. Any permissible left shift of an operation in Gantt chart cannot improve the makespan in the active schedule. There is a non-delay schedule, if no machine is kept idle after starting processing some operation in a schedule. Hence, the class of active schedules includes non-delay schedules.

There is a critical operation in a schedule, if the starting time of the operation cannot be delayed without deteriorating the makespan value. At least, one operation in a job or machine starts immediately after a critical one. Thus, the completion time of a critical operation is equal to the starting time of at least one of its successor operations. A critical operation cannot start delayed, since it has no buffer time available. The buffer time is the minimum time span between the completion time of an operation and maximum of starting time of its job successor and machine successor. For instance, consider o_{32} and o_{22}, which are direct successors of operation o_{21}. Here the buffer time is calculated as min(6, 7) – 4 = 2. A head can be defined as the earliest starting time of an operation and determines the amount of time needed before the operation can be started. Hence, a tail $q_{o_{ij}}$ gives the time needed for rest of the production schedule after completion of an operation. For all critical operations o_{ij}, head, processing time and tail are added to C_{max} as in Equation (3.5).

$$C_{max} = r_{o_{ij}} + P_{ij} + q_{o_{ij}} \tag{3.5}$$

From a viewpoint of an operation, a tail indicates the amount of time needed to complete a schedule. The tail gives an upper bound regarding the makespan in case of non-critical operations.

3.9 IMPROVEMENT OF SCHEDULE USING PERMUTATION

Makespan of a feasible schedule can be improved by interchanging two adjacent operations in a critical path, if these two operations are also adjacent within the same machine. For example, makespan represented by the schedule shown in Figure 3.5 is 17. Pair of adjacent operations in a critical path and in machines are (2, 7), (8, 3) and (3, 11) and interchanges of these pairs in the existing schedule produce the makespan values of 14, 15 and 16 respectively. Hence, the makespan value is improved from 17 to 14 by the application of the first pair (2, 7) belonging to second machine and the ultimate machine schedule is given as follows.

Sequence of operations in machine one :

$$\begin{tikzcd} 1 \arrow[r] & 10 \arrow[r] & 9 \arrow[r] & 6 \end{tikzcd}$$

Sequence of operations in machine two :

$$\begin{tikzcd} 7 \arrow[r] & 2 \arrow[r] & 5 \arrow[r] & 12 \end{tikzcd}$$

Sequence of operations in machine three :

$$\begin{tikzcd} 4 \arrow[r] & 8 \arrow[r] & 3 \arrow[r] & 11 \end{tikzcd}$$

Critical path of this schedule can be found similar to the method in section 3.6 and following critical path is generated.

$$\begin{tikzcd} \boldsymbol{s} \arrow[r, Rightarrow] & \mathbf{4} \arrow[r, Rightarrow] & 8 \arrow[r, Rightarrow] & \mathbf{3} \arrow[r, Rightarrow] & 11 \arrow[r, Rightarrow] & \mathbf{12} \arrow[r, Rightarrow] & 13 \arrow[r, Rightarrow] & \boldsymbol{e} \end{tikzcd}$$

Now pair of adjacent operations in the critical path and machines are (4, 8), (8, 3) and (3, 11) and these operations are all belonging to third machine. Interchanges of these pairs in the current schedule produce makespan values of 16, 15 and 13. Hence, the makespan value is improved from 14 to 13 by the application of the pair (3, 11) and new machine schedule is given below:

Sequence of operations in machine one :

$$\begin{tikzcd} \textcircled{1} \arrow[r] & \textcircled{10} \arrow[r] & \textcircled{9} \arrow[r] & \textcircled{6} \end{tikzcd}$$

Sequence of operations in machine two :

$$\begin{tikzcd} \textcircled{7} \arrow[r] & \textcircled{2} \arrow[r] & \textcircled{5} \arrow[r] & \textcircled{12} \end{tikzcd}$$

Sequence of operations in machine three :

$$\begin{tikzcd} \textcircled{4} \arrow[r] & \textcircled{8} \arrow[r] & \textcircled{11} \arrow[r] & \textcircled{3} \end{tikzcd}$$

Above schedule represents the makespan value of 13, which is the optimal one for the current problem. Hence, for small problems, the optimal makespan value can be easily found in the polynomial time. But for larger size problems, meta-heuristic methods are employed and are described and implemented in following chapters.

3.10 SUMMARY

This chapter has provided overview of job shop scheduling problems. The standard model of a job shop scheduling problems has been defined. The representation of the problems using disjunctive graph model has been presented. This chapter has focused on a framework for building a schedule from the scratch using the graph. Critical path in the graph has been found and Gantt chart has been used to represent the ultimate schedule. Permutation method has been employed to improve the constructed schedule.