4.2 MINIMIZATION OF MAKESPAN USING JOHNSON'S RULE FOR ($F_2 \| C_{max}$) PROBLEM

The flow shop contains **n** jobs simultaneously available at time zero and to be processed by two machines arranged in series with unlimited storage in between them. The processing times of all jobs are known with certainty. It is required to schedule the **n** jobs on the machines so as to minimize makespan (C_{max}). This problem is solved by Johnson's non-preemptive rule for optimizing the makespan in the general two machine static flow shop. This is the most important result for the flow shop problem which has now become a standard in theory of scheduling. The Johnson's rule for scheduling jobs in two machine flow shop is given below:

In an optimal schedule, job ***i*** precedes job ***j*** if:

$$\min \{p_{i1}, p_{j2}\} < \min \{p_{j1}, p_{i2}\}$$

Where as,

p_{i1} is the processing time of job ***i*** on machine 1 and p_{i2} is the processing time of job i on machine 2. Similarly, p_{j1} and p_{j2} are processing times of job j on machine 1 and 2 respectively.

The steps of Johnson's algorithm for constructing an optimal schedule may be summarized as follows:

Let,

p_{1j} = processing time of job j on machine 1.

p_{2j} = processing time of job j on machine 2.

Johnson's Algorithm

Step 1: Form set-I containing all the jobs with $p_{1j} < p_{2j}$

Step 2: Form set-II containing all the jobs with $p_{1j} > p_{2j}$

The jobs with $p_{1j} = p_{2j}$ may be put in either set.

Step 3: Form the sequence as follows:

a) The jobs in set-I go first in the sequence and they go in increasing order of p_{1j} (SPT)

b) The jobs in set-II follow in decreasing order of p_{2j} (LPT). Ties are broken arbitrarily.

This type of schedule is referred to as SPT (1)-LPT (2) schedule.

Example 4.1

Consider the following data presents an instance of $F_2 || C_{max}$ problem. Find optimal value of makespan using Johnson's rule.

Job (j)	j_1	j_2	j_3	j_4	j_5
p_{1j}	5	2	3	6	7
p_{2j}	1	4	3	5	2

Solution:

Step 1:

Of all jobs; $1 \leq j \leq 5$, only job j_2 has $p_{1j} < p_{2j}$ which belong to Set-I = { j_2 }

Step 2:

Jobs j_1, j_4 and j_5 have $p_{1j} > p_{2j}$ which belong to Set-II = { j_1, j_4 , j_5 }

Job j_3 has $p_{1j} = p_{2j}$, so put it in any set; say set-I. Set-I = { j_2 , j_3 }

Step 3:

a) Arrange sequence of jobs in set-I according to SPT. Set-I contains j_2 and j_3 as members. Process time of job 2 on machine M_1 is p_{12}=2. Similarly process time of job 3 on machine M_1 is p_{13}=3. Sequencing jobs j_2 and j_3 according to SPT;

Set-I = { j_2 , j_3 }

b) Arrange sequence of jobs in set-II according to LPT. Process times of jobs in set-II are; $p_{21} = 1$, $p_{24} = 4$ and, $p_{25} = 2$. Hence, revised sequence is;

Set-II = { j_4, j_5, j_1 }

Optimal sequence; Set-I + Set-II = { j_2, j_3, j_4, j_5, j_1}

The schedule for the optimal sequence is presented in graphical form using directed graph and Gantt chart. Directed graph also presents the critical path. All the processes on machine M_1 are on critical path. Gantt chart shows idle times on machine M_2.

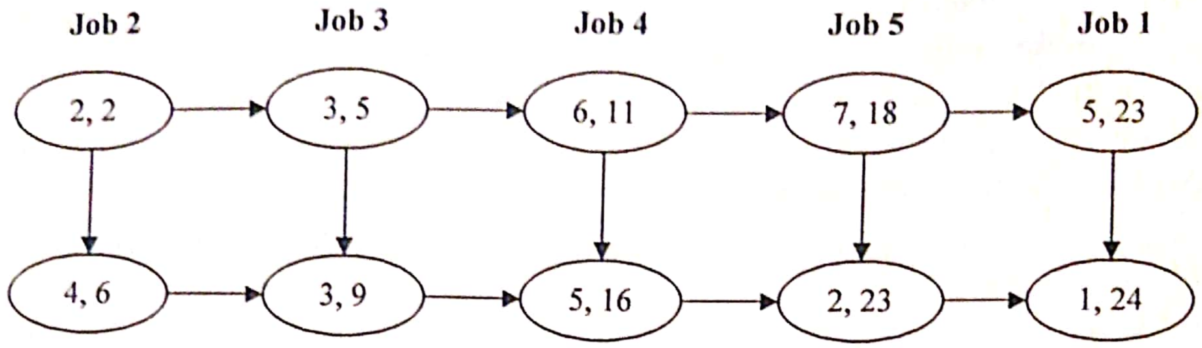

Figure 4.1 Directed Graph For Optimal Sequence { j_2, j_3, j_4, j_5, j_1}

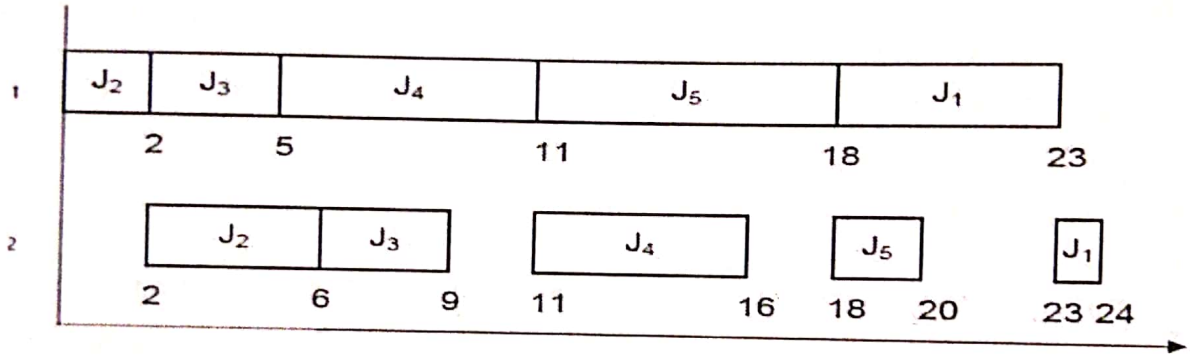

Figure 4.2 Gantt chart For optimal sequence { j_2, j_3, j_4, j_5, j_1}

4.3 MINIMIZATION OF MAKESPAN FOR ($F_3 \| C_{max}$) PROBLEM

This is the same flow shop problem as defined in two-machine case except that now there are three machines arranged in series for processing of n jobs in a prescribed order. Also, by virtue of dominance property number 2 (Theorem 2), permutation schedule still constitutes a dominant set for optimizing makespan. However, Johnson's 2-machine algorithm can not be extended to general 3-machine flow shop. Nevertheless, under special conditions, generalization is possible. In this regard, if either of the following condition satisfies, the Johnson's 2 machine algorithm may be extended to the 3-machine flow shop to achieve optimum makespan.

Either,

$$\min(p_{1j}) \geq \max(p_{2j})$$

or

$$\min(p_{3j}) \geq \max(p_{2j})$$

In other words, machine 2 is completely dominated by either the first or the third machine so that no bottleneck could possibly occur on the second machine. Subject to the above conditions, the optimal scheduling rule is applicable to 3-machine flow shop

The working procedure is the same as that described in the two machines case except that the three machines flow shop is reduced to two dummy machine M_1' and M_2' such that processing times of job j on machines M_1' and M_2' are $(p_{j1}+p_{j2})$ and $(p_{j2}+p_{j3})$ respectively. Johnson's algorithm is then applied to these two dummy machines to find the optimal job sequence.

Example 4.2

Consider an instance of the F3 || C_{max} problem in the following Table.

Job (j)	Process time (M1)	Process time (M2)	Process time (M3)
1	8	2	4
2	5	4	5
3	6	1	3
4	7	3	2

Find optimal sequence.

Solution:

Check for minimum value of process time on machines M_1 and M_3. These times are 5 and 2 respectively. Check maximum time on machine M_2 which is 4. Since min { p_{1j} } >= max { p_{2j}}, the problem can be converted to surrogate 2-machine problem. The problem data for two surrogate machines $M1'$ and $M2'$ is given in the following table.

Job (j)	Process time (M_1')	Process time (M_2')
1	10	6
2	9	9
3	7	4
4	10	5

Applying Johnson's Rule;

Set-I = { j_2 }, Set-II = { j_1, j_4, j_3 }

Optimal sequence = { j_2, j_1, j_4, j_3 }

Application of Johnson's algorithm to three machine flow shop problem has been tested by various authors. Burns and Rooker showed that under the conditions

$p_{j2} > \min(p_{j1}, p_{j3})$ for each job j=1,.....n, Johnson's algorithm produces optimal schedule. Jackson presented a case where all jobs use a common first and third machine for operation one and three respectively, but for second operation; the machine differs with each job. Under the conditions he observed that Johnson's algorithm produces optimal makespan for the 3-machine flow shop problem.

For general flow shops where the condition min $\{p_{1j}\}$ >= max $\{p_{2j}\}$ or min $\{p_{3j}\}$ >= max $\{p_{2j}\}$ is relaxed, Johnson's algorithm does not necessarily produce optimum makespan. However, it does provide good starting schedule, which can be further improved towards optimality through employing various techniques. In this context, Giglio and Wagner tested the algorithm for the series of the problems whereby the average makespan of 20 different cases under Johnson's Rule came out to be the 131.7 as compared to 127.9 for the optimal schedules. Furthermore, in 9 cases the true optimal results were obtained and another 8 results could be made optimum by interchanging the sequence of two adjacent jobs. Therefore, apparently Johnson's algorithm seems to produce good starting solutions, which even if not optimal, possesses the potential of heading towards optimality with reduced efforts.