UNIT - 8 SEQUENCING

The Sequencing Problem involves the determination of an optimal order or sequence of performing a series of jobs by a number of facilities (that are arranged in a specific order) so as to optimize the total time or cost.

Although, theoretically, it is always possible to select the best sequence by testing each one; in practice, it is impossible because of the large number of computations involved. For example, if there are 4 jobs to be processed on each of the 5 machines (ie, $n = 4$ and $m = 5$), the total number of theoretically possible different sequences will be $(4!)^5 = 7,962,624$.

The various optimality criteria normally resorted to are:

1. minimizing total elapsed time (or makespan)
2. minimizing mean flow time (or mean time in the job shop)
3. minimizing idle time of machines.
4. Minimizing total tardiness: Lateness of a job is defined as the difference between the actual

completion time of the job and its due date. If lateness is positive, it is termed as tardiness. Total tardiness is the sum of tardiness over all the jobs in the set.

5. minimizing number of tardy jobs.
6. Minimizing in-process inventory cost.
7. Minimizing the cost of being late.

A general sequencing problem may be defined as follows:

Let there be n jobs $(1, 2, 3, \ldots, n)$, each of which has to be processed, one at a time, on each of the m machines $(A, B, C, \ldots)$. The order of processing each job through the machines is given (for example, job 1 is processed on machines A, C, B, in this order). Also, the time required for processing each job on each machine is given. The problem is to find among $(n!)^m$ possible sequences, that technologically feasible sequence for processing the jobs which gives the minimum total elapsed time for all the jobs.

Elements of Sequencing Problem:

① Number of machines: It refers to no of service facilities through which a job must pass before it is completed.

②. Processing order: It refers to the order in which machines are required for completing the job.

③. Processing time: It is the time required by a job on each machine.

④ Idle time: It is the time for which a machine remains idle during the total elapsed time.

⑤ Total elapsed time: It is the time period from the start of processing of first job on the first machine to the completion of the last job on the last machine. It is equal to processing time plus idle time on each machine.

⑥ No passing rule: No passing means maintaining the order in which jobs are to be processed on two machines.

Analytic methods, have been developed for solving only five simple cases:

(i) n Jobs and one machine A.

(ii) n jobs and two machines A and B; all jobs processed in the order, say AB, other limitations described in section 5.4.

(iii) n jobs and three machines A, B and C; all jobs processed in the order, say ABC, other limitations described in section 5.5.

(iv). Two jobs and m machines; each job to be processed through the machines in a prescribed order, not necessarily the same for both jobs.

(v). n Jobs and m machines A, B, C, ..., K; all jobs processed in the order, say ABC....K, other limitations described in section 5.7.

ASSUMPTIONS IN SEQUENCING PROBLEMS

The following simplifying assumptions are usually made while dealing with sequencing problems:

(i) only one operation is carried out on a machine at a particular time.

(ii) each operation, once started, must be completed.

(iii) an operation must be completed before its succeeding operation can start.

(iv) only one machine of each type is available.

(v) a job is processed as soon as possible, but only in the order specified.

(vi) Processing times are independent of order of performing the operations.

(vii) The transportation time i.e., the time required to transport jobs from one machine to another is negligible.

(viii) Jobs are completely known and are ready for processing when the period under consideration starts.

(ix) The cost of in-process inventory for each job is same and negligibly small.

PROCESSING OF n JOBS THROUGH ONE MACHINE

Consider a static job shop wherein 'n' different jobs with known processing times require processing on a single machine. The job shop is static in the sense that any new job that arrives does not disturb the processing of these 'n' jobs. So it is assumed that new job arrivals wait for being considered in the next batch of jobs after the processing of the current 'n' jobs is completed.

Let

n = number of different jobs,

t_i = processing time of Job i,

w_i = waiting time (before processing) for job i,

F_i = flow time of job $i = w_i + t_i$,

C_i = Completion time of job i,

d_i = due date of job i,

L_i = lateness of job $i = C_i - d_i$,

E_i = earliness of job $i = d_i - C_i$,

T_i = tardiness of job i,

and N_T = number of tardy jobs.

A. Shortest Processing Time (SPT) Rule.

(a) Sequencing the jobs in a way that the job with least processing time is picked up first, followed by the one with the next smallest processing time and so on is known as SPT sequencing and achieves the following objectives:

(i) minimizing mean waiting time,

(ii) minimizing mean flow time,

(iii) minimizing mean lateness,

(iv) minimizing the mean number of jobs waiting as in-process inventory.

(b) In case importance of the jobs varies, a weight w_i is assigned to each job, a larger value indicating greater importance. Processing times are divided by the weights and jobs sequenced in order of increasing $\frac{t_i}{w_i}$.

The weighted mean flow time,

$$WMFT = \frac{\sum_{i=1}^{n} w_i F_i}{\sum_{i=1}^{n} w_i}$$

This rule is called weighted shortest processing time (WSPT) rule.

Problem 1

Eight jobs 1, 2, ..., 8 are to be processed on a single machine. The processing times, due dates and importance weights of the jobs are represented in table 5.1.

TABLE 5.1

Job	Processing time t_i (minutes)	Due date d_i (minutes)	Importance weight w_i	$\frac{t_i}{w_i}$
1	5	15	1	5.0
2	8	10	2	4.0
3	6	15	3	2.0
4	3	25	1	3.0
5	10	20	2	5.0
6	14	40	3	4.7
7	7	45	2	3.5
8	3	50	1	3.0

Assuming that no new jobs arrive thereafter, determi[ne] using SPT rule and WSPT rule

(i) Optimal Sequence

(ii) Completion time of the jobs.

(iii) mean flow time as well as weighted mean flow time

(iv) average in-process inventory.

(v) Lateness, mean lateness and maximum lateness,

(vi) number of jobs actually late.

<u>Solution</u>:

1. SPT Rule

(i) Optimal sequence is 4-8-1-3-7-2-5-6

(ii) Completion times of these jobs are:

3, 6, 11, 17, 24, 32, 42 and 56 minutes respectively.

(iii) Mean flow time $= \dfrac{3+6+11+17+24+32+42+56}{8} =$

$$= \frac{191}{8} = 23.875 \text{ minutes.}$$

(iv) Number of jobs waiting as in-process inventory are 8 during time 0-3, 7 during 3-6, 6 during 6-11, 5 during 11-17, 4 during 17-24, 3 during 24-32, 2 during 32-42 and 1 during 42-56 minutes

∴ Average in-process inventory.

$$= \frac{8\times3+7\times3+6\times5+5\times6+4\times7+3\times8+2\times10+1\times14}{3+3+5+6+7+8+10+14}$$

$= \frac{191}{56} = 3.41$ jobs.

(v) Lateness of the various jobs is given by:

Job no. :	4	8	1	3	7	2	5	6
Lateness : (minutes)	-22	-44	-4	2	-21	22	22	16

(3-25)

$$\text{mean lateness} = \frac{-22-44-4+2-21+22+22+16}{8}$$

$$= -\frac{29}{8} = -3.625 \text{ minutes.}$$

Maximum lateness = 22 minutes.

(vi) Number of jobs actually late = 4.

(2). WSPT Rule

(i) $\frac{t_i}{w_i}$ is calculated for each job and is shown in the last column of table 5.1. The optimal sequence as per non-decreasing $\frac{t_i}{w_i}$ values is 3-4-8-7-2-6-1-5.

Rest is similar to SPT calculation.

Earliest Due Date (EDD) Rule

According to this rule jobs are sequenced in the order of non-decreasing due dates. This rule minimizes the maximum job lateness as well as maximum job tardiness. However, this rule tends to make more jobs tardy and increases the mean tardiness.

Same problem | applying EDD Rule.

(i) optimal sequence as per EDD rule is

2-1-3-5-4-6-7-8.

Rest is similar to SPT calculation.

Slack Time Remaining (STR) Rule.

Slack time for a job is defined as the due date of the job minus its processing time. Sequencing the jobs in such a way that the jobs with the least slack time are picked up first for processing followed by the one with the next smallest slack time and so on is called the slack time remaining (STR) rule.

Example 2 Slack time remaining schedule.

Job	:	A	B	C	D	E	F	G
Processing time (days)	:	4	12	2	11	10	3	6
Due date (days)	:	20	30	15	16	18	5	9

Job	Processing time, ti (days)	Due date di (days)	Slack time (days) (di - ti)
A	4	20	16
B	12	30	18
C	2	15	13
D	11	16	5
E	10	18	8
F	3	5	2
G	6	9	3

Accordingly the STR schedule is $F \rightarrow G \rightarrow D \rightarrow E \rightarrow C \rightarrow A \rightarrow B$.

Also can do First come, first served (FCFS) schedule.

Processing n JOBS through Two machines

There are n different jobs to be processed on two machines and it is desired to determine the optimal sequence of jobs that minimizes T, the total elapsed time from the start of the first job on first machine to the completion of the last job on second machine. The total elapsed time includes the idle time, if any.

The following conditions are assumed:

(i) only two machines are involved, A and B.

(ii) Each job is processed in the order AB i.e., whichever job is processed first on machine A must also be processed first on machine B and so on; no passing being allowed.

(iii) Set-up times of machines A and B are independent of the sequence in which the jobs are taken up.

(iv) In-process storage space is available and the cost of in-process inventory is either same for each job or is too small to be considered. This, however, is correct only for processes involving short duration. For longer processes, inventory cost must also be considered.

(v) order of completion of jobs has no significance i.e., no job is required more urgently than the other.

(vi) The actual or expected processing times $A_1, A_2, \ldots, A_n$; $B_1, B_2, \ldots, B_n$ are known and represented by a table of the type shown below.

TABLE.

Machine times for n jobs and two machines.

Job i	A	B
1	A_1	B_1
2	A_2	B_2
3	A_3	B_3
⋮	⋮	⋮
i	A_i	B_i
⋮	⋮	⋮
n	A_n	B_n

It can be shown that the shortest elapsed time occurs when all jobs are processed on the two machines in the same order. The solution procedure given below (without proof) is due to S.M. Johnson and R. Bellman. It consists of the following steps:

Step 1: Examine the columns of processing times on machines A and B and find the smallest value [$\min(A_i, B_i)$].

Step 2: If this value falls in column A, schedule this job first on machine A. If this value falls in column B, schedule this job last on machine A (because of the given order AB). If there are equal minimal values (there is tie) one in each column, schedule the one in the first column first on machine A; and the one in the second column, last on machine A. If both equal values are in the first column (A), select the one with lowest entry in column B first. If the equal values are in the second column (B), select the one with the lowest entry in column A first.

Step 3 : Cross out the job assigned and continue the process (repeat steps 1 and 2), placing the jobs next to first or next to last till all the jobs are scheduled. The resulting sequence will minimize T.

Example 5.4-1

A machine operator has to perform two operations, turning and threading, on a number of different jobs. The time required to perform these operations (in minutes) for each job is known. Determine the order in which the jobs should be processed in order to minimize the total time required to turn out all the jobs.

Job	Time for turning (minutes)	Time for threading (minutes)
1	3	8
2	12	10
3	5	9
4	2	6
5	9	3
6	11	1

Also find the total processing time and idle time for turning and threading operations. [VTU June 2011]

Solution

The solution procedure is described below:

By examining the columns, we find the smallest value. It is threading time of 1 minute for Job 6 in second column. Thus we schedule Job 6 last for turning (and thereafter for threading) as shown below.

					6

The reduced set of processing times becomes

Job.	Turning time (minutes)	Threading time (minutes).
1	3	8
2	12	10
3	5	9
4	2	6
5	9	3

The smallest value is turning time of 2 minutes for job 4 in first column. Thus we schedule job 4 first as shown below.

4					6

The reduced set of processing times becomes

Job	Turning time (minutes)	Threading time (mins)
1	3	8
2	12	10
3	5	9
5	9	3

There are two equal minimal values: turning time of 3 minutes for job 1 in first column and threading time of 3 minutes for job 5 in second column. According to the rules, job 1 is scheduled next to job 4 and 5 next to job 6 as shown below

4	1			5	6

The reduced set of processing times becomes

Job	Turning time (minutes)	Threading time (mins)
2	12	10
3	5	9

The smallest value is turning time of 5 mins for job 3 in first column. Therefore, we schedule job 3, next to job 1 and we get the optimal sequence as.

4	1	3	2	5	6

Now we can calculate the elapsed time corres-ponding to the optimal sequence, using the individual processing times given in the problem. The details are shown in table 5.4.

TABLE 5.4

Job	Turning operation		Threading operation	
	Time in	Time out	Time in	Time out
4	0	2	2	8
1	2	5	8	16
3	5	10	16	25
2	10	22	25	35
5	22	31	35	38
6	31	42	42	43

Thus the minimum elapsed time is 43 minutes. Idle time for turning operation is 1 minute (from 42nd minute to 43rd minute) and for threading operation is 2 + 4 = 6 minutes (from 0-2 and 38-42 minutes).

Example 5.4-2

There are seven jobs, each of which has to go through the machines A and B in the order AB. Processing times in hours are given as.

Job :	1	2	3	4	5	6	7
Machine A :	3	12	15	6	10	11	9
Machine B :	8	10	10	6	12	1	3

Determine a sequence of these jobs that will minimize the total elapsed time T. Also find T and idle time for machines A and B.

						6

1					7	6

1	4				7	6

or

1				4	7	6

2 3

1	4	5	~~3~~	~~2~~	7	6

or

2 3

1	5	~~3~~	~~2~~	4	7	6

Job.	Machine A Time in	Machine A Time out	Machine B Time in	Machine B Time out	Idle for machine B
1	0	3	3	11	3
4	3	9	11	17	0
5	9	19	19	31	2
3	19	34	34	44	3
2	34	46	46	56	2
7	46	55	56	59	0
6	55	66	66	67	7

Thus the minimum elapsed time is 67 hours.

Idle time for machine A is 1 hour (66th-67th hour)

and for machine B is 17 hours.

PROCESSING n JOBS through THREE MACHINES

This sequencing problem is completely described as follows:

(i) only three machines A, B and C are involved,
(ii) each job is processed in the prescribed order ABC (first on machine A, then on B and thereafter on C)
(iii) no passing of jobs is permitted (ie., the same order over each machine is maintained), and
(iv) The actual or expected processing times $A_1, A_2, \ldots, A_n$; $B_1, B_2, \ldots, B_n$ and $C_1, C_2 \ldots C_n$ are known and represented by a table of the type shown below.

TABLE 5.10.
Machine times for n jobs and three machines

Job	A	B	C
1	A_1	B_1	C_1
2	A_2	B_2	C_2
3	A_3	B_3	C_3
⋮	⋮	⋮	⋮
i	A_i	B_i	C_i
⋮	⋮	⋮	⋮
n	A_n	B_n	C_n

The problem, again, is to find the optimum sequence of jobs which minimizes T.

No general solution is available at present for such a case. However, the method of section 5.4 can be extended to cover the special cases where either one or both of the following conditions hold good (if neither of the conditions holds good, the method fails and the optimal sequence has to be found by enumerating all the sequences).

(1) the minimum time on machine A is $\geq$ maximum time on machine B, and

(2) the minimum time on machine C is $\geq$ maximum time on machine B.

The method, described here without proof, is to replace the problem by an equivalent problem involving n jobs and two machines. These two (fictitious) machines are denoted by G and H and their corresponding processing times are given by

$$G_i = A_i + B_i, \quad H_i = B_i + C_i.$$

If this new problem with the prescribed order GH is solved by the method of section 5.4, the resulting optimal sequence will also be optimal for the original problem.

Example 5.5-1

A machine operator has to perform three operations turning, threading and knurling on a number of different jobs. The time required to perform these operations (in minutes) for each job is known. Determine the order in which the jobs should be processed in order to minimize the total time required to turn out all the jobs. Also find the idle times for the three operations

TABLE 5.11

Job	Time for turning (minutes)	Time for threading (minutes)	Time for knurling (minutes)
1	3	8	13
2	12	6	14
3	5	4	9
4	2	6	12
5	9	3	8
6	11	1	13

Solution:

Here, min $A_i = 2$, max $B_i = 8$, and min $C_i = 8$.

Since min C_i = max B_i, we can solve this problem by the procedure described in section 5.3.

The equivalent problem involving 6 jobs and two fictitious operations G and H becomes

Processing times for 6 jobs & 2 fictitious operations

Job	G_i = Turning + Threading (minutes)	H_i = Threading + Knurling (minutes)
1	11	21
2	18	20
3	9	13
4	8	18
5	12	11
6	12	14

Examining the columns G_i and H_i, we find that the smallest value is 8 under operation G_i in row 4. Thus we schedule job 4 first (on operation G_i and thereafter on H_i) as shown below.

4					

The reduced set of processing times becomes

Job	G_i	H_i
1	11	21
2	18	20
3	9	13
5	12	11
6	12	14

The next smallest value is 9 under column G_i for Job 3. Hence we schedule job 3 as shown below.

4	3				

The reduced set of processing times becomes

Job	G_i	H_i
1	11	21
2	18	20
5	12	11
6	12	14

There are two equal minimal values: processing time of 11 minutes under column G_i for Job 1 and processing time of 11 minutes under column H_i for Job 5. According to the rules, job 1 is scheduled next to job 3 and 5 is scheduled last as shown below.

4	3	1			5

The reduced set of processing times becomes

Job	G_i	H_i
2	18	20
6	12	14

The smallest value is 12 under column G_i for Job 6. Hence we schedule job 6 next to job 1 and the optimal sequence becomes.

4	3	1	6	2	5

Now we may calculate the elapsed time corresponding to the optimal sequence, using the individual processing times given in the problem. The details are shown in table 5.12.

TABLE 5.12

Job	Turning Operation Time in	Turning Operation Time out	Threading Operation Time in	Threading Operation Time out	Knurling operation Time in	Knurling operation Time out
4	0	2	2	8	8	20
3	2	7	8	12	20	29
1	7	10	12	20	29	42
6	10	21	21	22	42	55
2	21	33	33	39	55	69
5	33	42	42	45	69	77

Thus the minimum elapsed time is 77 minutes. Idle time for turning operation is $77 - 42 = 35$ minutes, for threading operation is $2+1+11+3+(77-45) = 17+32 = 49$ minutes and ~~for~~ knurling operation is 8 minutes.

PROCESSING TWO JOBS THROUGH m MACHINES

This sequencing problem is described as follows.

(a) There are m machines, denoted by A, B, C, ..., K,

(b) only two jobs are to performed: job 1 and job 2,

(c) the technological ordering of each of the two jobs through m machines is known. This ordering may or may not be the same for both jobs. Alternative ordering is not permissible for either job.

(d) The actual or expected processing times $A_1, B_1, C_1, \dots, K_1; A_2, B_2, C_2, \dots, K_2$ are knowns

(e) each machine can work only one job at a time and storage space for in-process inventory is available.

The problem is to minimize the total elapsed time T i.e., to minimize the time from the start of first job to the completion of the second job.

Such a problem can be solved by graphical method which is simple and provides good (though not necessarily optimal) results.

Example 1.

Using graphical method, determine the optimal sequence needed to process jobs 1 and 2 on five machines, A, B, C, D and E. For each machine find the job which should be done first. Also calculate the total time needed to complete both the jobs.

TABLE.

Job 1	Sequence :	A	B	C	D	E
	Time (hrs) :	1	2	3	5	1
Job 2	Sequence :	C	A	D	E	B
	Time (hrs) :	3	4	2	1	5

Soln:

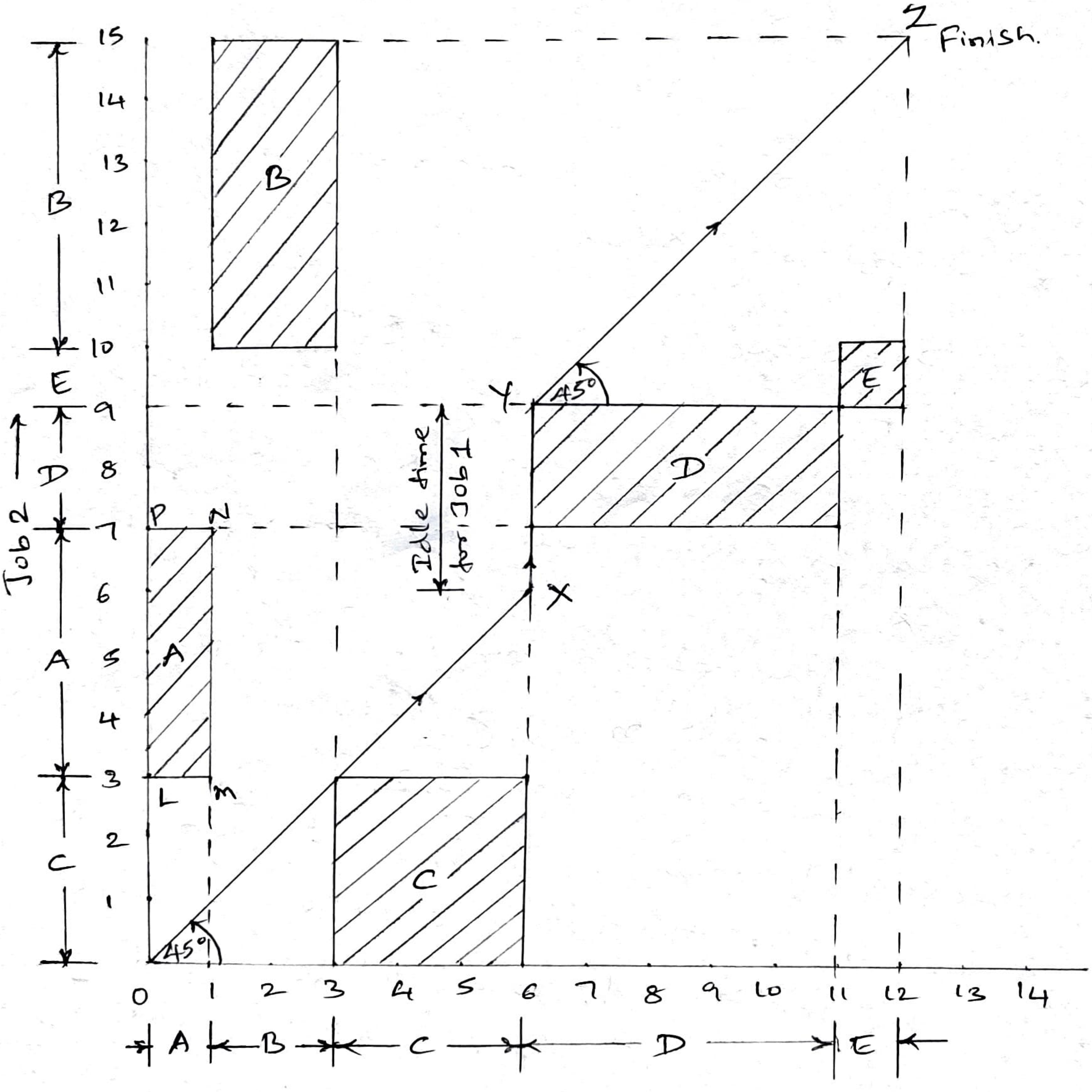
The graphical procedure is described with the help of the following steps;

Step 1: Draw two axes at right angles to each other. Represent processing time on job 1 along horizontal axis and processing time on job 2 along vertical axis. Scale used must be same for both the jobs.

Step 2: Layout the machine times for the two jobs on corresponding axes in the given technological order. This is shown in figure 3.2.

Step 3: Machine A requires 1 hour for job 1 and 4 hours for job 2. A rectangle LMNP is, thus, constructed for machine A. Similar rectangles are constructed for machines B, C, D and E as shown.

Step 4: Make a program by starting from origin (O) and moving through the various stages of completion (points) till the point marked 'finish' is reached. Choose path consisting only of horizontal, vertical and 45° lines. A horizontal line represents work on job 1 while job 2 remains idle, a vertical line represents work on job 2 while job 1 remains idle and a 45° line to the base represents simultaneous work on both jobs.



Graphic Solution of 2 job and 5 machine problem.

Step 5: Find the optimal path (program). An optimal path is one that minimizes idle time for job 1 (vertical movement). Likewise, an optimal path is one that minimizes idle time for job 2 (horizontal movement). Obviously, the optimal path is one which coincides with 45° line to the maximum extent.

Further, both jobs cannot be processed simultaneously on one machine. Graphically, this means that diagonal movement through the blocked out areas is not allowed.

A good path, accordingly, is chosen by eye and drawn on the graph (path OXYZ).

Step 6: Find the elapsed time. It is obtained by adding the idle time for either job to the processing time for that job. The idle time for the chosen path is found to be 3 hours for Job 1.

∴ Total elapsed time = 12+3 = 15 hours (considering job 1)
= 15+0 = 15 hours (considering job 2).

Step 7: The optimal sequence corresponding to the chosen path is shown in Fig. 5.3.

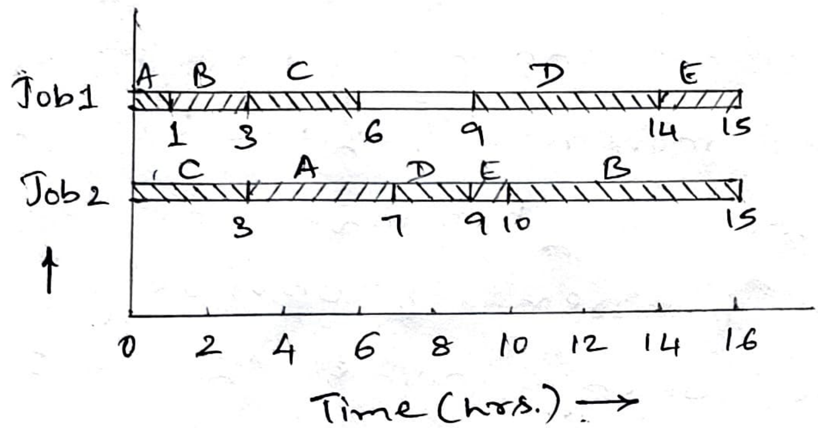


Fig 5.3

The optimal sequence or schedule on various machines for the two jobs as evident from Fig 5.3 is

Machine A : Job 1 precedes Job 2,

Machine B : Job 1 precedes Job 2,

Machine C : Job 2 precedes Job 1,

Machine D : Job 2 precedes Job 1,

and Machine E : Job 2 precedes Job 1.

PROCESSING n JOBS THROUGH m MACHINES.

This sequencing problem is described as follows:

(i) There are n jobs to be performed, denoted by 1, 2, 3, ... i, ... n.

(ii) there are m machines, denoted by A, B, C, ... K

(iii) each Job is to be processed in the prescribed order, say ABC...K.

(iv) no passing of Jobs is permitted (i.e., same order over each machine is maintained).

(v) the actual or expected processing times $A_1, A_2 \ldots A_n$, $B_1, B_2, \ldots, B_n$; $C_1, C_2, \ldots, C_n \ldots$, $K_1, K_2, \ldots K_n$ are known and represented by a table of the type shown below.

TABLE.

Machine times for n jobs and m machines

Job	A	B	C	...	K
1	A_1	B_1	C_1	--	K_1
2	A_2	B_2	C_2	--	K_2
3	A_3	B_3	C_3	--	K_3
⋮	⋮	⋮	⋮	⋮	⋮
i	A_i	B_i	C_i	--	K_i
⋮	⋮	⋮	⋮	⋮	⋮
n	A_n	B_n	C_n	--	K_n.

The problem, as before, is to find the optimum sequence of jobs which minimizes T.

Case I

method of section 5.5 can be applied (extended) to cover the special cases where either one or both of the following conditions hold good (if neither of the conditions holds good, the method fails):

(i) the minimum time on machine A is $\geq$ maximum time on machines $B, C, \ldots, K-1$,

(ii) the minimum time on machine K is $\geq$ maximum time on machines $B, C, \ldots K-1$.

The method is to replace the m machine problem by an equivalent two machine problem. These two (fictitious) machines are denoted by a and b, and their corresponding processing times are given by

$$a_i = A_i + B_i + \dots + (K-1)_i,$$

$$b_i = B_i + C_i + \dots + (K-1)_i + K_i$$

If this new problem with the prescribed order ab is solved by the method of section 5.5, the resulting optimal sequence will also be optimal for the original problem.

Further, if

$$B_i + C_i + \dots + (K-1)_i = k,$$

where k is a fixed positive constant for all jobs ($i = 1, 2, 3, \dots, n$) then the given problem can be solved simply as n job two machine problem (where the two machines are A and K in the order AK) as per the method of section 5.4.

Example 5.7-1

Four jobs 1, 2, 3 and 4 are to be processed on each of the five machines A, B, C, D and E in the order ABCDE. Find the total minimum elapsed time if no passing of jobs is permitted. Also determine idle time for each machine.

TABLE.

M/C Job	A	B	C	D	E
1	7	5	2	3	9
2	6	6	4	5	10
3	5	4	5	6	8
4	8	3	3	2	6

Solution

Here, min $A_i = 5$, min $E_i = 6$ and max (B_i, C_i, D_i) $= 6, 5, 6$ respectively.

Since min E_i = max (B_i, D_i), we can solve this problem by the procedure described in section 5.7.

The equivalent problem involving 4 jobs and 2 fictitious machines a and b becomes.

Job	machine a	machine b
1	17	19
2	21	25
3	20	23
4	16	14

Examining the columns we find that the optimal sequence is.

1	3	2	4

Now we may calculate the total elapsed time corresponding to the optimal sequence using the individual processing times given in the original problem. The details are shown in the table.

TABLE.

M/C Job	A	B	C	D	E
1	0-7	7-12	12-14	14-17	17-26
3	7-12	12-16	16-21	21-27	27-35
2	12-18	18-24	24-28	28-33	35-45
4	18-26	26-29	29-32	33-35	45-51

Thus the minimum elapsed time is 51 time units.

Idle time on machine

$A = 51 - 26 = 25$ time units,

$B = 7 + 2 + 2 + (51 - 29) = 33$ time units,

$C = 12 + 2 + 3 + 1 + (51 - 32) = 37$ time units,

$D = 14 + 4 + 1 + (51 - 35) = 35$ time units.

and $E = 17 + 1 = 18$ time units.

CASE II

Sequencing problems involving n jobs and m machines wherein neither of the conditions described in caseI earlier are satisfied can be solved by the procedure described below.

Step 1: The given n×m problem is split into a number of n×2 subproblems. The number of such problems will be m-1. Thus a 3-machine problem will involve 3-1=2 subproblems a 4-machine problem will involve 4-1=3 subproblems and so on.

For example, a 4-machine problem involving machines A, B, C and D will yield the following 2-machine subproblems:

1. Involving machines A & D
2. Involving machines A+B and C+D
3. Involving machines A+B+C and B+C+D.

Step 2: Each 2-machine subproblem is solved as per the method of section 5.4.

Step 3: All the solutions are examined. The sequence that involves the least processing time or cost is the optimal.

Example 5.7-4

Five Jobs 1, 2, ... 5 are to be processed on four machines A, B, C and D. Their processing times are given in table 5.25. Determine the optimal sequence, minimum elapsed time and idle time for each machine.

TABLE 5.25

Processing times in hours

Job	A	B	C	D
1	7	15	14	21
2	11	18	18	6
3	2	13	11	16
4	14	4	27	14
5	18	11	32	16

Solution:

Step 1: Here min $A_i = 2$, max $B_i = 18$, max $C_i = 32$, min $D_i = 6$. Since neither of the conditions are satisfied, the problem is split up into the following three 2-machine subproblems

Subproblem 1			Subproblem 2			Subproblem 3		
Job	Processing times on machines		Job	Processing times on machines		Job	Processing times on machines	
	A	D		A+B	C+D		A+B+C	B+C+D
1	7	21	1	22	35	1	36	50
2	11	6	2	29	24	2	47	42
3	2	16	3	15	27	3	26	40
4	14	14	4	18	41	4	45	45
5	18	16	5	29	48	5	61	59

Step 2: Optimal sequence is now determined for each subproblem. They are

1. For Subproblem 1: 3 → 1 → 4 → 5 → 2
2. For Subproblem 2: 3 → 4 → 1 → 5 → 2
3. For Subproblem 3: 3 → 1 → 4 → 5 → 2

Step 3: Total processing time is now calculated for each different sequence,

Sequence 3 → 1 → 4 → 5 → 2

Sequence	A	B	C	D
3	0-2	2-15	15-26	26-42
1	2-9	15-30	30-44	44-63
4	9-23	30-34	44-71	71-85
5	23-41	41-52	71-103	103-119
2	41-52	52-70	103-121	121-127

∴ Total elapsed time is 127 hours,

Sequence 3 → 4 → 1 → 5 → 2

Sequence	A	B	C	D
3	0-2	2-15	15-26	26-42
4	2-16	16-20	26-53	53-67
1	16-23	23-38	53-67	67-88
5	23-41	41-52	67-99	99-115
2	41-52	52-70	99-117	117-123

∴ Total elapsed time is 123 hours.

Hence optimal sequence: 3 → 4 → 1 → 5 → 2,

minimum elapsed time: 123 hours,

Idle time for machine A: 123 − 52 = 71 hours

Idle time for machine B: 2 + 1 + 3 + 3 + (123 − 70)
= 62 hours,

Idle time for machine C: 15 + (123 − 117) = 21 hours

and Idle time for machine D: 26 + 11 + 11 + 2 = 50 hours