5

Sequencing Models And Related Problems

This chapter deals with the situations in which the effectiveness measure (time, cost, distance, etc.) is a function of the order or sequence of performing a series of jobs (tasks). The selection of the appropriate order in which waiting customers may be served is called *sequencing*. Sequencing problems can be classified in two groups:

1. In the first group, there are n jobs to be performed, where each job requires processing on some or all of m different machines. The order in which these machines are to be used for processing each job as well as the expected or actual processing time of each job on each of the machines is known. We can also measure the effectiveness for any given sequence of jobs at each of the machines and we wish to select from the $(n!)^m$ *theoretically feasible* alternatives, the one which is both *technologically feasible* and optimizes the effectiveness measure (*e.g.*, minimizes the total elapsed time from the start of the first job to the completion of the last job as well as idle time of machines). A technologically feasible sequence is one which satisfies the constraints (if any) on the order in which each job must be performed through the m machines. The technology of manufacturing processes renders many sequences technologically infeasible. For example, a part must be degreased before it is painted; similarly, a hole must be drilled before it is threaded.

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 at each of the 5 machines (*i.e.*, $n = 4$ and $m = 5$), the total number of theoretically possible different sequences will be $(4!)^5 = 7{,}962{,}624$. Of course, as already said, some of them may not be feasible because the required operations must be performed in a specified order. Obviously, any technique which helps us arrive at an optimal (or at least approximately so) sequence without trying all or most of the possibilities will be quite valuable. .

2. The second group of problems deals with job shops having a number of machines and a list of tasks to be performed. Each time a task is completed by a machine, the next task to be started on it has got to be decided. Thus the list of tasks will change as fresh orders are received.

Unfortunately, both types of problems are intrinsically difficult. While solutions are possible for some simple cases of the first type, only some empirical rules have been developed for the second type till now.

5.1. Sequencing Problems

In sequencing problems, there are two or more customers to be served (or jobs to be done) and one or more facilities (machines) available for this purpose. We want to know when each job is to begin and what its due date is. We also want to know which facilities are required to do each job, in which order these facilities are required and how long each operation is to take.

Sequencing problems have been most commonly encountered in production shops where different products are to be processed over various combinations of machines.

However, sequencing problems can arise even where only one service facility is involved, for example, a number of programs waiting to get on a computer or a number of patients waiting for a doctor.

A general sequencing problem may be defined as follows:

Let there be n jobs (1, 2, 3, ..., n), each of which has to be processed, one at a time, on each of m machines (A, B, C, ...). 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.

Symbolically,

Let A_i = time required for job i on machine A,
B_i = time required for job i on machine B, etc., and
T = total elapsed time for jobs 1, 2, ..., n *i.e.*, time from start of the first job to completion of the last job.

The problem is to determine a sequence $(i_1, i_2, ..., i_n)$ where $(i_1, i_2, ..., i_n)$ is a permutation of integers (1, 2, ..., n) which will minimize T.

Analytic methods have been developed for solving only four simple cases:

(1) n jobs and two machines A and B; all jobs processed in the order AB; other limitations described in section 5.2.

(2) n jobs and three machines A, B and C; all jobs processed in the order ABC, other limitations described in section 5.3.

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

(4) n jobs and m machines A, B, C, ..., K; all jobs processed in the order ABC ... K, other limitations described in section 5.5.

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.

5.2. Processing *n* Jobs through two Machines

This sequencing problem is completely described as follows:

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

(*ii*) each job is processed in the order AB, and

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

Table 5.1

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

The problem is to determine the sequence (order) of jobs which minimizes T, the total elapsed time from the start of first job to the completion of last job.

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 for 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.

Some important assumptions made while following the above solution procedure are

(*i*) it is assumed that the order of completion of jobs has no significance *i.e.*, no product is required more urgently than the other.

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

EXAMPLE 5.2-1

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.

Table 5.2

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

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 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 (minutes)*
1	3	8
2	12	10
3	5	9
5	9	3