1 Introduction

Sequencing is the process of scheduling jobs on machines in such a way so as to minimize the overall time, cost and resource usage thereby maximizing profits.

Let,
\[ n \rightarrow \text{Number of jobs} \]
\[ m \rightarrow \text{Number of machines} \]
Then, theoretically possible sequences = \( (n!)^m \)
e.g. if \( n = 4 \) and \( m = 3 \)
then
Number of sequences = \( (4!)^3 = 16,777,216 \)

Not possible practically to check out all sequences to find the optimal one. Hence the need of finding some suitable procedure for sequencing the jobs.

PRIORITY SEQUENCING

When jobs compete for work centres capacity, which job should be done next? Priority sequence rules are applied to all jobs waiting in the queue. Then when the work centre becomes open for the job, the one with the highest priority is assigned. ”Priority Sequencing” is a systematic procedure for assigning priorities to waiting jobs thereby determining the sequence in which the jobs will be performed.

SEQUENCING PROBLEMS

The sequencing problem arises whenever there is a need for determining an optimum order of performing a number of jobs by number of facilities according to some pre-assigned order so as to optimise the output in terms of cost, time or profit.

Production of finished goods from raw materials consists of several operations to be performed in a given sequence. Frequently, similar operations required for several products are performed at the same work stations particularly in intermittent or batch production. Under such situations, it is required to select a preferred order for products passing through a work station. The problem becomes complicated when the several work stations serve many products. In such a problem, the criteria is minimum total processing time.
The general sequencing problem is stated as: There are n jobs (1, 2, 3...n) each of which must be processed through each of m machines (m₁, m₂, m₃, ....mₙ) one at a time. The order of processing each job through the machines is given and also the time taken to process each job on each machine is known.
The problem is to determine the order of processing of n jobs so that the total elapsed time for all the jobs will be minimum. The general sequencing problem is to determine the optimal sequence from amongst \((n!)^m\) sequences that minimises the total elapsed time.

Basic Assumptions:

1. Only one operation is carried out on a machine at a time.
2. Processing times are known and do not change.
3. Processing times are independent of order of processing the job.
4. The time required in moving jobs from one machine to another is negligibly small.
5. Each operation once started must be performed till completion.
6. Each preceding operation must be completed, before beginning of the next immediate operation.
7. Only one machine of each type is available.
8. A job is processed as soon as possible, but only in the order specified.
9. ‘No passing rule’ is strictly followed. i.e. same order of jobs is maintained over each machine. e.g. If n jobs are to be processed on two machines A → B, then each job should go to machine A first then to machine B.

2 Sequencing Models

All type of sequencing problems may be categorized in one of the following models:

- Sequencing n jobs on 1 machine.
- Sequencing n jobs on 2 machine.
- Sequencing n jobs on 3 machine.
- Sequencing n jobs on m machine.
2.1 Sequencing n jobs on 1 machine

Five rules to find out optimal sequence:

1. SPT rule (Shortest Processing Time).
2. WSPT rule (Weighted Shortest Processing Time).
3. EDD rule (Earliest Due Date).
4. Hodgson’s Algorithm.
5. Slack Rule.

To illustrate above rules an example is being considered:

Example: Consider 8 jobs with processing times, due dates and importance weights as shown below:

<table>
<thead>
<tr>
<th>Job(i)</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processing time ($t_i$)</td>
<td>5</td>
<td>8</td>
<td>6</td>
<td>3</td>
<td>10</td>
<td>14</td>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>Due Date ($d_i$)</td>
<td>15</td>
<td>10</td>
<td>15</td>
<td>25</td>
<td>20</td>
<td>40</td>
<td>45</td>
<td>50</td>
</tr>
<tr>
<td>Importance Weight ($w_i$)</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

2.1.1 SPT Rule

In this rule, job with shortest processing time is considered first then next and so on. It simply means that arranging processing time in ascending order, the job sequence could be found.

Using SPT rule the sequence of jobs will be 4-8-1-3-7-2-5-6

Completion of these jobs are at times 3, 6, 11, 17, 24, 32, 42, 56 respectively.

a) Mean Flow Time = \( \frac{3+6+11+17+24+32+42+56}{8} = \frac{191}{8} \) = 23.9 units

b) Weighted Mean Flow Time = \( \frac{1 \times 3 + 1 \times 6 + 1 \times 11 + 3 \times 17 + 2 \times 24 + 2 \times 32 + 2 \times 42 + 3 \times 56}{1+1+1+3+2+2+2+3} = \frac{435}{15} = 29 \) units

c) Average in-process inventory
Average in-process inventory = \( \frac{8 \times 3 + 7 \times 3 + 6 \times 5 + 5 \times 6 + 4 \times 7 + 3 \times 8 + 2 \times 10 + 1 \times 14}{56} = \frac{191}{56} = 3.41 \) jobs

d) Waiting time for each job:

<table>
<thead>
<tr>
<th>Job(i):</th>
<th>4</th>
<th>8</th>
<th>1</th>
<th>3</th>
<th>7</th>
<th>2</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Waiting time:</td>
<td>0</td>
<td>3</td>
<td>6</td>
<td>11</td>
<td>17</td>
<td>24</td>
<td>32</td>
<td>42</td>
</tr>
</tbody>
</table>

Mean Waiting time = \( \frac{9+3+6+11+17+24+32+42}{8} = \frac{135}{8} \) = 16.875 units
e) Lateness of each job = Completion time of job - Due date of job

Lateness of job 4 = 3 - 25 = -22
Lateness of job 8 = 6 - 50 = -44
Lateness of job 1 = 11 - 15 = -4
Lateness of job 3 = 17 - 15 = +2
Lateness of job 7 = 24 - 45 = -21
Lateness of job 2 = 32 - 10 = +22
Lateness of job 6 = 56 - 40 = +16

Positive Lateness is called tardiness.

Mean Lateness = \(-22 -44 -4 +2 +21 +22 +22 +22 +16\) \(\frac{8}{8}\) = -3.625 units

Mean tardiness = 9 + 0 + 0 + 2 + 0 + 22 + 22 + 16 \(\frac{8}{8}\) = 7.8 units

No. of tasks actually late = 4 (job no. 3,2,5,6) = No. of tardy jobs

Maximum lateness = 22 units (job 5 and 2 both)

### 2.1.2 WSPT Rule

In this rule, job with shortest ‘processing time per unit of importance’ is considered first then next and so on. It simply means that arranging ‘processing time per unit of importance’ in ascending order, the job sequence could be found.
Using WSPT rule the sequence of jobs will be 3-4-8-7-2-6-1-5

Respective Flowtime = 6, 9, 12, 19, 27, 41, 46, 56

Mean Flow Time = \[
\frac{6+9+12+19+27+41+46+56}{8} = \frac{216}{8} = 27 \text{ units}
\]

b) Weighted Mean Flow Time = \[
\frac{3\times6+1\times9+1\times12+2\times19+2\times27+3\times41+1\times46+2\times56}{3+1+1+2+2+3+1+2} = \frac{412}{15} = 27.47 \text{ units}
\]

Similarly, Mean Lateness = -0.5

Mean Tardiness = 10.6

No. of Tardy Jobs = 4 (job No. 2,6,1,5)]

Max. Tardiness = 36 units

### 2.1.3 EDD Rule

In this rule, jobS are sequenced in the order of increasing due dates of jobs.

Using EDD rule the sequence of jobs will be 2-1-3-5-4-6-7-8

Mean Flow Time = 32

Weighted Mean Flow Time = 31.7

Mean Lateness = +4.5 units

Mean Tardiness = +5 units

Maximum Lateness = 9 units

No. of late jobs = 6 (job no. 3.5.4.6.7.8) = No. of tardy jobs

### 2.1.4 Hodgson’s Algorithm

This algorithm is aplicable only if the number of tardy jobs is more than 1. Using EDD rule number of tardy jobs could be found.

As per EDD rule, the sequence of jobs will be 2-1-3-5-4-6-7-8 with 6 tardy jobs which is more than 1 job. Hence Hodgson’s algorithm is applicable.
Since job 3 is first tardy job and is in 3rd position, examine first 3 jobs to identify the one with longest processing time. Job 2 has longest processing time of 8 units. Hence remove it and make the table again.

5th job is tardy. Since longest processing time up to 5th job i.e. from 1st, 3rd and 5th job is that of 5th job, we will remove it.

Now as per Hodgson’s algorithm, the sequence of jobs will be 1-3-4-6-7-8-2-5

Mean Lateness = 1.625
Mean Tardiness = 9
Max. Lateness = 36 units (both 2nd and 5th job)
Number of tardy jobs = 2 (job 2 and 5)

### 2.1.5 Slack Rule

Slack time = Due time of job - its processing time

This rule is based on sequencing jobs in ascending order of slack time.

**Job(i):**

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>3</th>
<th>5</th>
<th>4</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
<td>10</td>
<td>3</td>
<td>14</td>
<td>7</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>15</td>
<td>6</td>
<td>10</td>
<td>3</td>
<td>14</td>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>11</td>
<td>24</td>
<td>38</td>
<td>45</td>
<td>48</td>
<td></td>
</tr>
</tbody>
</table>

**Due Date (d_i):**

<table>
<thead>
<tr>
<th></th>
<th>15</th>
<th>15</th>
<th>20</th>
<th>25</th>
<th>40</th>
<th>45</th>
<th>50</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
<td>15</td>
<td>20</td>
<td>25</td>
<td>40</td>
<td>45</td>
<td>50</td>
</tr>
<tr>
<td>3</td>
<td>15</td>
<td>24</td>
<td>38</td>
<td>45</td>
<td>45</td>
<td>50</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>24</td>
<td>38</td>
<td>45</td>
<td>45</td>
<td>50</td>
<td></td>
</tr>
</tbody>
</table>

**Slack time (d_i - t_i):**

<table>
<thead>
<tr>
<th></th>
<th>10</th>
<th>2</th>
<th>9</th>
<th>22</th>
<th>10</th>
<th>26</th>
<th>38</th>
<th>47</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-10</td>
<td>-4</td>
<td>-1</td>
<td>-2</td>
<td>0</td>
<td>-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>-10</td>
<td>-4</td>
<td>-1</td>
<td>-2</td>
<td>0</td>
<td>-2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>-10</td>
<td>-4</td>
<td>-1</td>
<td>-2</td>
<td>0</td>
<td>-2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Taking slack times in ascending order, the sequence of jobs will be 2-3-1-5-4-6-7-8

Completion times are 8,14,19,29,32,46,53 & 56
Mean tardiness = 5 units
Mean Flow time = 32.1 units
Weighted Mean Flow time = 31.1 units
2.2 Sequencing n jobs on 2 machine

The problem is defined as,
There are only two machines A and B. Each job is processed in the order A and B.
The processing times of n jobs (1, 2, 3 ...... n) on each of the two machines is known. Let
A₁, A₂,.....Aₙ, are processing times on A and B₁, B₂,.....Bₙ are processing times on B.
The problem is to find the order in which the n jobs are to be processed to minimise the
total elapsed time (T) to complete all the n jobs.

Procedure using Johnson’s Algorithm
Step 1: Select the smallest processing time from the given list of processing times A₁, A₂,.....Aₙ
and B₁, B₂,.....Bₙ.
Step 2: If the minimum processing time is Aᵣ (i.e., Job number r on Machine A), do the
ᵣth job first in the sequence. If the minimum processing time is Bₛ (i.e., job number s on
Machine B), do the sth job last in the sequence.
Step 3: After doing this step, (n-1) jobs are left to be sequenced. Repeat step (1) and step
(2) till all the jobs are ordered.
Step 4: Find the total processing time as per the sequence determined and also determine
idle time associated with machines.

2.3 Sequencing n jobs on 3 machine

There are three machines M1, M2 and M3. Each job has to go through three machines in
the order M1, M2 and M3.
Conditions to be satisfied to solve the above problem by Johnson’s method:
1. The smallest processing time on machine M1 ≥ largest processing time on machine M2.
2. The smallest processing time on machine M3 ≥ largest processing time on machine M2.

If either or both of the above stated conditions are satisfied, the given problem can be solved by Johnson’s algorithm.

**Procedure using Johnson’s Algorithm**

Step I: Convert the three machine problem into two machine problem by introducing two fictitious machines G and H such that

\[ G_i = M_{1i} + M_{2i} \]
\[ H_i = M_{2i} + M_{3i} \text{ (where } i = 1, 2, 3, \ldots n) \]

Step II: Once the problem is converted to n job 2 machine the sequence is determined using Johnson’s algorithm for n Jobs and 2 machines.

Step III: For the optimal sequence determined, find out the minimum total elapsed time and idle times associated with machines.

**Tie breaking Rules**

1. If there are equal smallest processing times one for each machine, place the job on machine 1, first in the sequence and one in machine 2 last in the sequence.
2. If the equal smallest times are both for machine 1, select the job with lower processing time in machine 2 for placing first in the sequence.
3. If the equal smallest times are both from machine 2, select the one with lower processing time in machine 1, for placing last in the sequence.

**2.4 Processing n jobs through m machines**

Let there be n jobs which are to be processed through m machines \( M_1, M_2, \ldots, M_m \) in the order \( M_1, M_2, \ldots, M_m \).

Let \( T_{ij} \) denote the time taken by ith job on the jth machine.

**Procedure**

Step I: Find Min \( T_{i1} \) (Minimum time for the first machine).

Min \( T_{im} \) (Minimum time on the last machine).

Max \( (T_{ij}) \) For \( j = 2, 3, \ldots, m-1 \) and \( i = 1, 2, \ldots, n \) (Maximum time on intermediate machines).

Step II: Check for the following conditions. (i) Minimum time \( T_{i1} \) for the first machine \( M_1 \) ≥ Maximum time \( (T_{ij}) \) on intermediate machines \( (M_2 \text{ to } M_{m-1}) \)

(ii) Minimum time \( T_{im} \) for the last machine \( M_m \) ≥ Maximum time \( (T_{ij}) \) on intermediate machines \( (M_2 \text{ to } M_{m-1}) \)

i.e., the minimum processing time on the machines \( M_1 \) and \( M_m \) (First and last machines) should be ≥ maximum time on any of the 2 to \( m-1 \) machines.

Step III: If the conditions in step II are not satisfied, the problem cannot be solved by this
method, otherwise go to next step.

Step IV: Convert the $n$ job $m$ machine problem into $n$ job 2 machine problem by considering two fictitious machines $G$ and $H$. Such that,

\[ TG_{ij} = T_{i1} + T_{i2} + \ldots + T_{im} \]

\[ TH_{ij} = T_{i2} + T_{i3} + \ldots + T_{im} \]

Step V: Now obtain the sequence for $n$ jobs using Johnson’s Algorithm.

Step VI: Determine the minimum total elapsed time and idle times associated with machines.