Basic Scheduling Algorithms for Single Machine Problems

Single machine model is the simplest type of scheduling models and a special case of all other environments. It is often found in practice when there is only one service point or a single stage manufacturing.

Algorithms developed for the single machine model provide a basis for design of exact algorithms and heuristics for more complicated machine environments: multi-machine systems can often be decomposed into a number of single stage systems.

In this lecture, we focus on the classical single machine problems with objective functions ΣC_j, $\Sigma w_j C_j$.

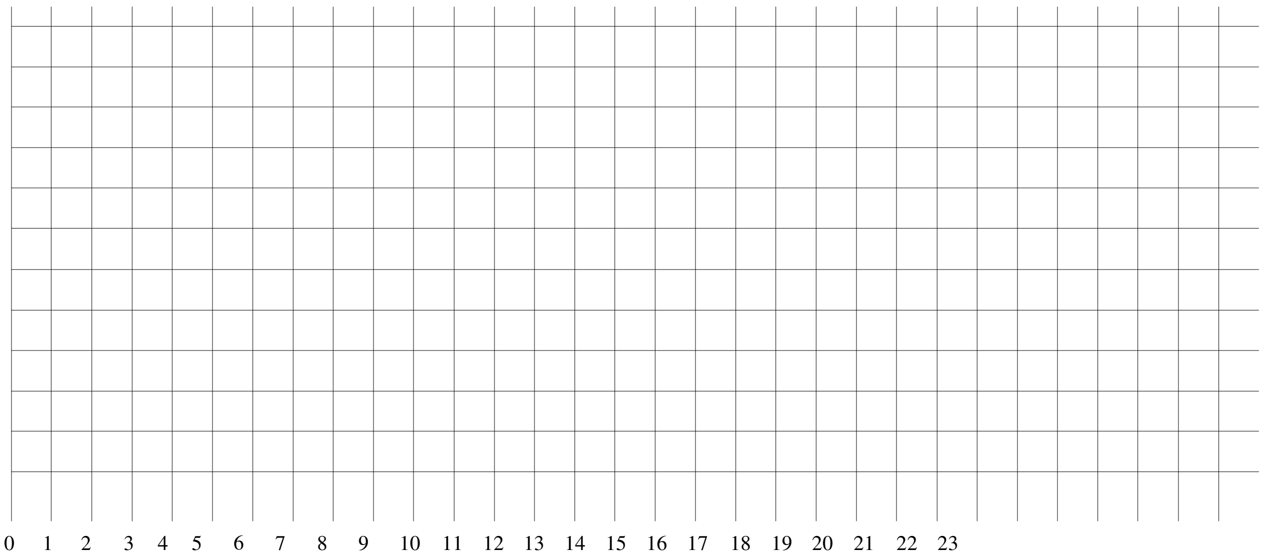
1. Minimising total completion time: $1||\Sigma C_j$

First let us develop an algorithm that finds an optimal schedule for $1||\Sigma C_j$.

Solve the following instance with n=3 jobs.

Job	p_j
1	7
2	2
3	5

To find its solution, let us generate all possible permutations of jobs and compute the completion times for the resulting schedules.



Formulate the rule that solves the problem: ____

In the example above, the jobs in the optimal schedule are arranged in increasing (non-decreasing) order of their processing times. The rule that sorts the jobs in this order is known as ***SPT*** (shortest processing time), i.e., the next job to be scheduled will be the job with the smallest processing time.

In general, take n jobs in order (1, 2, …, n) and compute their completion times:

$$
\begin{array}{lcccccccccc}
C_1 & = & p_1 & & & & & & & & \\
C_2 & = & p_1 & + & p_2 & & & & & & \\
 & \vdots & & & & & & & & & \\
C_{n-1} & = & p_1 & + & p_2 & + & \dots & + & p_{n-1} & & \\
C_n & = & p_1 & + & p_2 & + & \dots & + & p_{n-1} & + & P_n \\
\hline
\sum C_j & = & np_1 & + & (n-1)p_2 & + & \dots & + & 2p_{n-1} & + & P_n
\end{array}
$$

Thus, job 1 contributes np_1, job 2 contributes $(n-1)p_2$, and so on.

If we want to minimize $\sum C_j$, we want p_1 to be the smallest, p_2 the second smallest, etc.

Thus, to find an optimal solution to problem $1| |\ \Sigma C_j$ we need to sort the jobs in SPT order. As any other sorting, this requires O($n$log n) time.

2. Minimising total weighted completion time: $1||\Sigma\ w_jC_j$

Consider $1|| \Sigma w_jC_j$.
Here

- w_j is the importance of job j,
- Σw_jC_j characterises the total holding, or inventory costs incurred by the schedule.

We start with the following instance of problem $\mathbf{1|p_j=1|\Sigma\ w_jC_j}$ with n=3 jobs of unit length.

Job	p_j	w_j
1	1	7
2	1	2
3	1	5

Find an optimal sequence ____

and the optimal value of the objective function ______________________________

If all processing times are equal (p_j=1) then the problem can be solved by ________________________

____

For problem $1||\sum w_jC_j$ intuitively we want to combine the rule above with SPT.

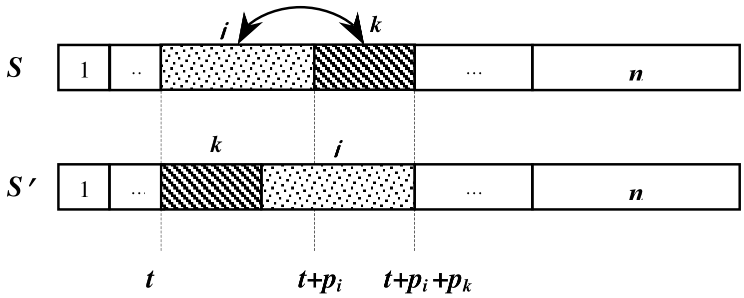
It seems logical to sort the jobs in non-decreasing order of the ratios $\boldsymbol{p_j / w_j}$. We call this rule **WSPT** (weighted SPT) rule or **Smith's rule** due to W.E. Smith who introduced it in 1956.

We can prove

Theorem 1. For $1|| \Sigma w_jC_j$, the WSPT rule is optimal.

Proof (adjacent pairwise interchange argument)

Suppose a schedule S, which is not WSPT, is optimal. In this schedule there must be at least two adjacent jobs i and k such that $p_i / w_i > p_k / w_k$. We assume that i precedes k and i starts at time t.



Swapping jobs i and k leads to a schedule S'.

We compare the value of the objective function $F = \Sigma w_jC_j$ for schedules S and S':

$$F(S) = \sum_{j\neq i,k} w_j C_j + w_i C_i + w_k C_k = \sum_{j\neq i,k} w_j C_j + w_i (t + p_i) + w_k (t + p_i + p_k)$$

$$F(S') = \sum_{j\neq i,k} w_j C_j + w_i C_i' + w_k C_k' = \sum_{j\neq i,k} w_j C_j + w_k (t + p_k) + w_i (t + p_i + p_k)$$

$$F(S') - F(S) = w_i p_k - w_k p_i < 0.$$

This contradicts the optimality of schedule S and completes the proof of the theorem. ■