**Basic Scheduling Algorithms for Single Machine Problems:**

**Processing Jobs With Preemption**

Consider a problem of scheduling jobs with preemption. In this problem, processing of any job can be interrupted and resumed later, the total processing time of all parts of job \( j \) being equal to \( p_j \).

The following exercise demonstrates that if all jobs are released at the same time \((r_j=0)\), there is no advantage to process the jobs with preemption. In other words, for problem \( 1|\text{pmtn}||\sum C_j \) there exists an optimal schedule without preemption.

An instance of problem \( 1|\text{pmtn}||\sum C_j \) with \( n=5 \) jobs is given by job processing times \( p_1 = 3; \ p_2 = 13; \ p_3 = 4; \ p_4 =2; \ p_5 =5. \)

Schedule \( S \) is represented by the Gantt chart below.

Construct schedule \( S' \) without preemption and without increasing completion times of all jobs.

Consider the general problem \( 1|\text{pmtn}||\sum C_j \) with \( n \) jobs and arbitrary processing times. To show that there exists an optimal nonpreemptive schedule describe a transformation that modifies an arbitrary preemptive schedule \( S \) into a nonpreemptive schedule \( S' \) without increasing the completion times of all jobs.
Minimising total completion time with nonzero release dates and preemption allowed: $1|r_j, pmtn|\sum C_j$

Consider $1|r_j, pmtn|\sum C_j$.

*Shortest Remaining Processing Time* (SRPT) rule: each time that a job is completed, or at the next release date, the job to be processed next has the smallest remaining processing time among the available jobs.

**Example:**

<table>
<thead>
<tr>
<th>Job</th>
<th>$r_j$</th>
<th>$p_j$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>13</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>14</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>18</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>22</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>25</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>30</td>
<td>4</td>
</tr>
<tr>
<td>8</td>
<td>33</td>
<td>1</td>
</tr>
</tbody>
</table>

For problem $1|r_j, pmtn|\sum C_j$, the SRPT rule is optimal.

**Proof** (pairwise interchange argument)

Consider a schedule in which available job $i$ with the shortest remaining processing time is not being processed at time $t$, and instead available job $k$ is being processed. Let $p_i'$ and $p_k'$ denote the remaining processing times for jobs $i$ and $k$ after time $t$, so $p_i' < p_k'$.

In total $p_i' + p_k'$ is spent on jobs $i$ and $k$ after time $t$. We assume that $C_i < C_k$. 

---

Single machine problems: preemption
Interchange:

1) Take the first $p'_i$ units of time that were devoted to either of jobs $i$ and $k$ after time $t$, and use them instead to process job $i$ to completion.

2) Take the remaining $p'_k$ units of time that were spent processing $i$ and $k$ after time $t$, and use them to schedule job $k$.

We have obtained a ‘better’ schedule $S'$:

$$C'_i < C_i$$

$$C'_k = C_k$$

Since in the new schedule all jobs other than $i$ and $k$ have the same the completion times as before, we obtain:

$$\sum_{j=1}^{n} C'_j - \sum_{j=1}^{n} C_j = (C'_i + C'_k) - (C_i + C_k) < 0.$$  

This contradicts the optimality of schedule $S$.  

$\blacksquare$