**LPT rule**: Whenever a machine becomes free for assignment, assign that job whose processing time is the largest among those jobs not yet assigned.

Example:

<table>
<thead>
<tr>
<th>Ji</th>
<th>J1</th>
<th>J2</th>
<th>J3</th>
<th>J4</th>
<th>J5</th>
<th>J6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pi</td>
<td>6</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>m1</th>
<th></th>
<th>3</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>J1</td>
<td></td>
<td></td>
<td>J6</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>m2</th>
<th></th>
<th></th>
<th>J2</th>
<th></th>
<th>J5</th>
</tr>
</thead>
<tbody>
<tr>
<td>J1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>m3</th>
<th></th>
<th></th>
<th>J3</th>
<th></th>
<th>J4</th>
</tr>
</thead>
<tbody>
<tr>
<td>J1</td>
<td></td>
<td>J2</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Optimal schedule:

\[ \sum p_i = 20 \]

\[ \sum p_i / m = 20 / 3 = 6.67 \]

Optimal schedule length must be an integer.

**Thm**: Let TS be a set of n independent jobs and m be the number of machines. We have

\[ \frac{t_{LPT}(TS, m)}{t_{opt}(TS, m)} \leq \frac{4}{3} - \frac{1}{3m}, \]

where \( t_{LPT}(TS, m) \) and \( t_{opt}(TS, m) \) denote the Cmax of the LPT schedule and the optimal schedule, respectively. Moreover, the bound is achievable for every m.
m = 2
3, 3, 2, 2, 2

LPT
0 3 5 7
3 2 2
3 2

OPT
0 6
2 2 2
3 3

m = 3
5, 5, 4, 4, 3, 3, 3

LPT
5 3 3 11
5 3
4 4

OPT
5 4
5 4
3 3 3

General m 2m+1 jobs

P_1 = P_2 = 2m - 1
P_3 = P_4 = 2m - 2
P_5 = P_6 = 2m - 3

\vdots

P_{2m-1} = P_{2m} = P_{2m+1} = m
For \( m \) even, we have

\[
\begin{align*}
\text{LPT} & \quad 3m-1 & \quad 4m-1 \\
J_1 & | & J_{2m} & & J_{2m+1} \\
J_2 & | & J_{2m-1} & & \\
J_3 & | & J_{2m-2} & & \\
J_4 & | & J_{2m-3} & & \\
\vdots & | & \vdots & & \\
J_{m-1} & | & J_{m+2} & & \\
J_m & | & J_{m+1} & & \\
\end{align*}
\]

\[
\begin{align*}
\text{OPT} & \quad 3m \\
J_1 & | & J_{2m-2} & & \\
J_2 & | & J_{2m-3} & & \\
J_3 & | & J_{2m-4} & & \\
J_4 & | & J_{2m-5} & & \\
\vdots & | & \vdots & & \\
J_{m-1} & | & J_m & & \\
J_{2m-1} & | & J_{2m} & | & J_{2m+1} \\
\end{align*}
\]
Proof: The theorem is trivially true for \( m = 1 \). Assume \( m \geq 2 \).

The proof is by contradiction.

1. Assume TS is the smallest (in terms of number of jobs) set of jobs that violates the bounds. Assume the jobs are ordered in descending order of their processing times, i.e. \( p_1 \geq p_2 \geq \ldots \geq p_n \).

Let \( s_i(S_{LPT}) \) and \( f_i(S_{LPT}) \) denote the starting time and finishing time of \( J_i \) in \( S_{LPT} \), respectively. Note that \( s_i(S_{LPT}) \leq s_{i+1}(S_{LPT}) \) for \( 1 \leq i \leq n \).

2. Claim 1: 
   1. \( f_n(S_{LPT}) = t_{LPT}(TS,m) \)
   2. \( f_i(S_{LPT}) < t_{LPT}(TS,m) \quad 1 \leq i \leq n-1 \)

Proof: Suppose not. Let \( J_r(r<n) \) be such that \( f_r(S_{LPT}) = t_{LPT}(TS,m) \). Consider \( TS' = \{J_1, \ldots, J_r\} \). Clearly \( t_{LPT}(TS',m) = t_{LPT}(TS,m) \). On the other hand, \( t_{OPT}(TS',m) \leq t_{OPT}(TS,m) \). Therefore,

\[
\frac{t_{LPT}(TS',m)}{t_{OPT}(TS',m)} \geq \frac{t_{LPT}(TS,m)}{t_{OPT}(TS,m)} > 4/3 - 1/3m
\]

Therefore \( TS' \) is a smaller set of jobs that violates the bound.

Contradiction!
Claim 2: In an optimal schedule for TS, no machine can execute more than 2 jobs.

Proof:

\[
\begin{align*}
t_{\text{OPT}}(\text{TS}, m) & \geq \frac{1}{m} \sum_{j=1}^{n} p_j \\
S_n(S_{\text{LPT}}) & \leq \frac{1}{m} \sum_{j=1}^{n-1} p_j \\
t_{\text{LPT}}(\text{TS}, m) & = S_n(S_{\text{LPT}}) + p_n \\
& \leq \frac{1}{m} \sum_{j=1}^{n-1} p_j + p_n \\
& = \frac{1}{m} \sum_{j=1}^{n} p_j + (1 - \frac{1}{m}) p_n
\end{align*}
\]

Therefore,

\[
\frac{t_{\text{LPT}}(\text{TS}, m)}{t_{\text{OPT}}(\text{TS}, m)} \leq \frac{1}{m} \sum_{j=1}^{n} p_j (j=1 \text{ to } n) + (1 - \frac{1}{m}) p_n ) / t_{\text{OPT}}(\text{TS}, m) \\
\leq 1 + [(1 - \frac{1}{m}) p_n / t_{\text{OPT}}(\text{TS}, m)]
\]

Since, \( t_{\text{LPT}}(\text{TS}, m) / t_{\text{OPT}}(\text{TS}, m) > 4/3 - 1/(3m) \), we have

\[
1 + (1 - \frac{1}{m}) p_n / t_{\text{OPT}}(\text{TS}, m) > 4/3 - 1/(3m) \\
or t_{\text{OPT}}(\text{TS}, m) < 3 p_n
\]

Since \( p_n \) is the smallest processing time, the optimal schedule can have at most 2 jobs on each machine.
We will transform the optimal schedule in such a way that the schedule length (makespan) will never be increased. Finally, it is shown that the transformed schedule is isomorphic to the LPT schedule. $\rightarrow \leftrightarrow$ since we assume that

$$\frac{t_{LPT}(TS, m)}{t_{OPT}(TS, m)} > \frac{4}{3} - \frac{1}{3}m > 1$$

**Type I operation**

$$\begin{align*}
M_k & \quad \begin{array}{c c c}
\text{Pi} & \text{Pj} & \text{Mk} \\
\end{array} \\
\downarrow & \\
M_k & \quad \begin{array}{c c c}
Pj & \text{Pi} & \text{No change in schedule length} \\
\end{array}
\end{align*}$$

**Type II operation**

$$\begin{align*}
M_k & \quad \begin{array}{c c c}
\text{Pi} & \text{Pi}' & \text{Mk} \\
\end{array} \\
\downarrow & \\
M_l & \quad \begin{array}{c c c}
Pj & \text{Pi} & \text{Ml} \\
\end{array} \\
\downarrow & \\
\end{align*}$$
Apply Types I, II, and III operations exhaustively until none can be applied. We obtain the following type of schedule (by rearranging machines if necessary)
Since no Type I operation can be applied, we have
\[ P_{i_l} \geq P_{j_l} \quad \text{for} \quad l \leq l \leq t \]
Since no Type II operation can be applied, we have
\[ P_k \geq P_{i_l} \]
Since no Type III operation can be applied, we have
\[ P_{j_l} \geq P_{j_{l-1}} \geq \ldots \geq P_{j_1} \]

Therefore \( P_{k_1} \geq P_{k_2} \geq \ldots \geq P_k \geq P_{i_l} \geq \ldots \geq P_i \geq P_{j_t} \geq \ldots \geq P_{j_1} \)

The transformed schedule looks very much like the LPT schedule. In fact, they can be different only in the assignment of the second layer jobs.
Consider 2 machines, both of which have 2 jobs assigned

\[ \begin{array}{c|c|c} \text{p} & P_i_p & P_j_p \\ \hline \text{q} & P_i_q & P_{j_q} \\ \end{array} \]

LPT rule would assign \( P_{j_p} \) to machine q.

But this means that there is an optimal schedule with 1 machine having 3 jobs.

\[ \rightarrow \leftarrow \text{ to claim 2.} \]

Hence, we conclude that the transformed schedule is isomorphic to \( S_{LPT} \). Therefore

\[ t_{LPT}(TS,m) = t_{OPT}(TS,m). \rightarrow \leftarrow \]

To show that at most a finite number of Type I, II, or III operations can be applied.

Define \( I_j(S) \) to be the last time \( t \) such that machine \( j \) becomes idle in \( S \).

Let \( J(S) = \sum_{1 \leq i < j \leq m} | I_i(s) - I_j(s) | \)

1. If \( S' \) is obtained from \( S \) by a Type I operation, then

\[ J(S') = J(S) \]
2. If $S'$ is obtained from $S$ by a Type II or III operation, then $J(S') < J(S)$

Between any Type II or III operations, there are at most $m$ Type I operation that can be applied. Only a finite number of Type II or III operations can be applied because of conditions (2) and because there are only a finite number of schedules.

\[
P \mid \text{prmp} \mid \text{Cmax}
\]

\[
P_2 \mid \text{prmp, prec} \mid \text{Cmax}
\]

\[
P \mid \text{prmp, intree} \mid \text{Cmax}
\]

\[
P \mid \text{prmp, outtree} \mid \text{Cmax}
\]

** All the above can be solved in polynomial time **

\[
P \mid \text{prmp, prec} \mid \text{Cmax} \text{ is NP-hard}
\]

\[
P_m \mid \text{prmp, prec} \mid \text{Cmax} \text{ is open for fixed } m > 2
\]
P | prmp | Cmax can be solved by McNaughton’s Rule

1. Compute $t = \max \{ \max \{P_i\}, \frac{1}{m} \sum_{i=1}^{n} P_i \}$

   $t$ is a lower bound for the optimal makespan

2. Schedule jobs one by one (wrap around if needed)