Section 2.3: Scheduling to minimize makespan

Makespan Scheduling on Identical Machines

Input:

m machines

n jobs w. processing times $p_1, \dots, p_n \in \mathbb{Z}^+$

Output:

Assignment of jobs to machines s.t. the makespan is minimized

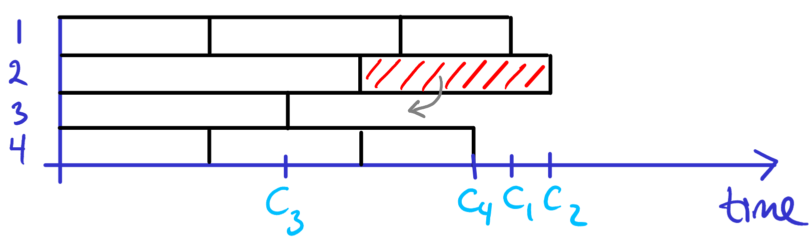
time when last machine finishes processing

Ex:

Input: 4, 5, 3, 8, 5, 6, 4, 4, 3

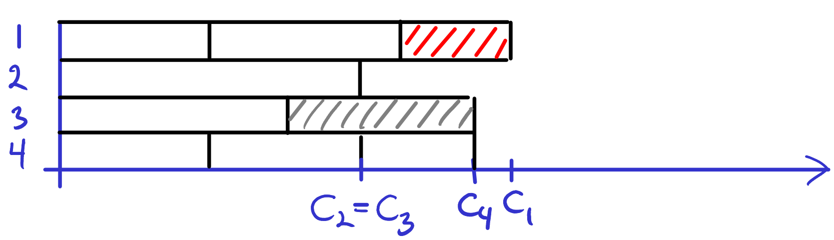
Output:

Machine no:



makespan = $\max\{C_1, C_2, C_3, C_4\} = C_2 = 13$

The schedule can be improved:



makespan = $C_1 = 12$

Local Search Alg.

Repeat
  job $\ell \leftarrow$ job that finishes last
  If $\exists$ machine i where job ℓ would finish earlier
    Move job ℓ to machine i
Until job ℓ is not moved

Theorem 2.5

The local search alg. is a $(2-\frac{1}{m})$-approx. alg.

Proof:

Let $p_{max} = \max_{1 \leq j \leq n} p_j$ and $P = \sum_{j=1}^{n} p_j$

Lower bounds on OPT:

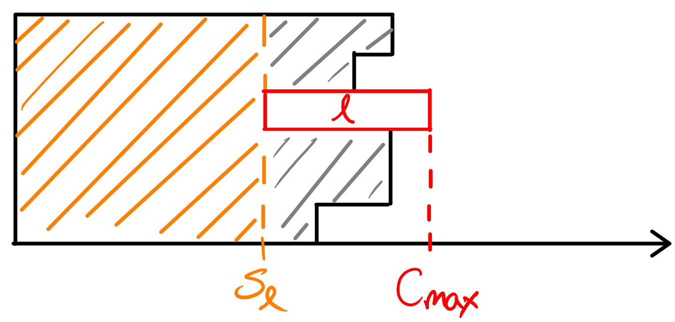
$OPT \geq p_{max}$ (*)

since the machine i with the longest job p_j has $C_i \geq p_j$

$OPT \geq \frac{P}{m}$ (**)

since this is the average completion time of the machines.

Upper bound on alg.'s makespan:



$$P \geq m \cdot S_\ell + p_\ell \quad \text{, since all machines are busy until } S_\ell$$

$$\Updownarrow$$

$$S_\ell \leq \frac{P - p_\ell}{m} \qquad (***)$$

$$\begin{aligned}
C_{max} &= S_\ell + p_\ell \\
&\leq \frac{P - p_\ell}{m} + p_\ell \text{, by } (***) \\
&= \frac{P}{m} + \left(1 - \frac{1}{m}\right) p_\ell \\
&\leq OPT + \left(1 - \frac{1}{m}\right) OPT \text{, by } (*) \text{ and } (**) \\
&= \left(2 - \frac{1}{m}\right) OPT
\end{aligned}$$

$\square$

What would be a natural greedy algorithm?

List Scheduling (LS)

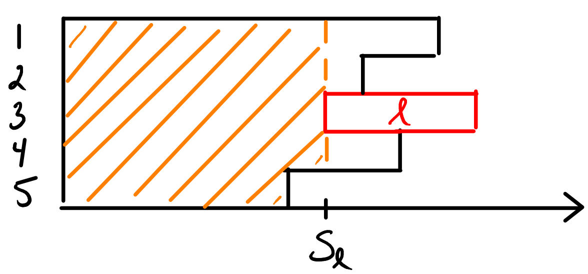
For $j \leftarrow 1$ to n
  Schedule job j on currently least loaded machine

Approx. ratio?

What properties of the local search alg. did we use to prove $2-\frac{1}{m}$?

We used only the fact that all machines are busy at least until S_ℓ (this was enough to prove (~~(***)~~)).

This is also true for LS:



LS would not have placed job ℓ on machine 3.

Theorem 2.6: LS is a $(2-\frac{1}{m})$-approx. alg.

Note that $\frac{LS}{OPT} < 2-\frac{1}{m}$, unless $p_\ell = p_{max}$ and all other machines finish by the time job ℓ starts.

Thus, it seems advantageous to schedule short jobs last.

Longest Processing Time (LPT)

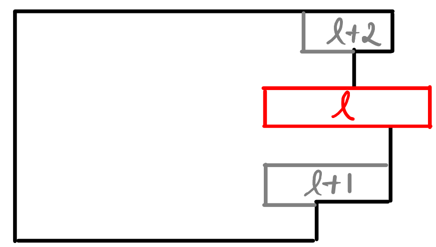
For each job j, in order of decreasing processing times
Schedule job j on currently least loaded machine

Theorem 2.7: LPT is a $\left(\frac{4}{3} - \frac{1}{3m}\right)$-approx. alg.

Proof:

Number the jobs s.t. $p_1 \geq p_2 \geq \cdots \geq p_n$.
(Then, the indices indicate the order in which the jobs are scheduled.)

Let job ℓ be a job to finish last:



We can assume that $\ell = n$:

Let $I = \{p_1, \ldots, p_n\}$ and $I_\ell = \{p_1, \ldots, p_\ell\}$.

Then, $LPT(I) = LPT(I_\ell)$, since jobs $\ell+1, \ldots, n$ finish no later than job ℓ.

Moreover, $OPT(I) \geq OPT(I_\ell)$, since $I_\ell \subseteq I$.

Thus, proving $\frac{LPT(I_\ell)}{OPT(I_\ell)} \leq \frac{4}{3} - \frac{1}{3m}$ will imply

$$\frac{LPT(I)}{OPT(I)} \leq \frac{LPT(I_\ell)}{OPT(I_\ell)} \leq \frac{4}{3} - \frac{1}{3m}.$$

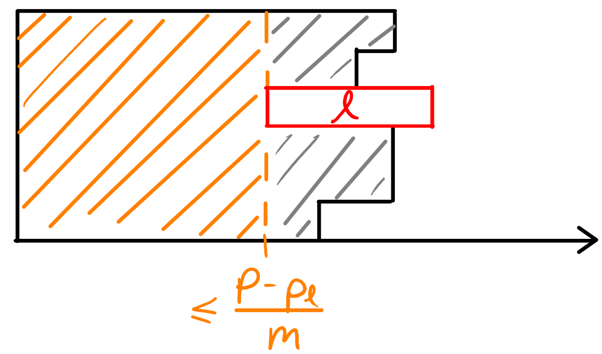
(Or said in a different way, we can ignore the jobs $\ell+1, \ldots, n$.)

Thus, we can assume that no job is shorter than job ℓ. This will be used in Case 2 below.

Case 1: $p_\ell \le \frac{1}{3} \cdot OPT$

Similarly to the proof of Thm 2.5:

$$
\begin{aligned}
LPT &\le \frac{P-p_\ell}{m} + p_\ell \\
&= \frac{P}{m} + \left(1-\frac{1}{m}\right) p_\ell \\
&\le OPT + \left(1-\frac{1}{m}\right) p_\ell \\
&\le OPT + \left(1-\frac{1}{m}\right) \cdot \frac{1}{3} OPT \\
&= \left(\frac{4}{3} - \frac{1}{3m}\right) OPT
\end{aligned}
$$



Case 2: $p_\ell > \frac{1}{3} \cdot OPT$

In this case, all jobs are longer than $\frac{1}{3}$ OPT. Hence, in OPT's schedule, each machine has at most 2 jobs, i.e., $n \le 2m$.

Claim: In this case $LPT = OPT$.

Proof of claim: Exercise 2.2.

$\square$

From the proof of Thm. 2.7, we learned:

- If $p_\ell > \frac{1}{3}\text{OPT}$, $\text{LPT} = \text{OPT}$.
- Otherwise, $\text{LPT} < \frac{4}{3}\text{OPT}$.

Can we balance the two cases better?

What if we first schedule all jobs of length at least $\frac{1}{4}\text{OPT}$ optimally, and then use LPT for the remaining jobs? What approx. factor would be obtained?

Would the schedule of the $\underbrace{\text{long jobs}}^{\text{length} \geq \frac{1}{4}\text{OPT}}$ have to be optimal to achieve this approx. factor?

From the proof of Thm. 2.7, we learned:

- If $p_\ell > \frac{1}{3}\mathrm{OPT}$, $\mathrm{LPT} = \mathrm{OPT}$.
- Otherwise, $\mathrm{LPT} \le \frac{4}{3}\mathrm{OPT}$.

Can we balance the two cases better?

What if we first schedule all jobs of length at least $\frac{1}{4}\mathrm{OPT}$ optimally, and then use LPT for the remaining jobs?

If the last job to finish is a long job,

$$C_{max} = \mathrm{OPT},$$

since the long jobs are scheduled optimally.

Otherwise,

$$\begin{aligned} C_{max} &\le \mathrm{OPT} + \left(1-\frac{1}{m}\right)p_\ell, \quad \text{by the proof of Thm. 2.5} \\ &\le \mathrm{OPT} + \left(1-\frac{1}{m}\right)\cdot\frac{1}{4}\cdot\mathrm{OPT} \\ &< \frac{5}{4}\cdot\mathrm{OPT} \end{aligned}$$

Would the schedule of the long jobs have to be optimal?

No, a $\frac{5}{4}$-approx. would suffice:

If the last job to finish is a long job,

$$C_{max} \le \frac{5}{4}\cdot\mathrm{OPT}$$

Otherwise,

$$C_{max} < \mathrm{OPT} + p_\ell \le \frac{5}{4}\cdot\mathrm{OPT}.$$

This sketches the idea for a PTAS ...

1. Schedule long jobs ($> \varepsilon \cdot OPT$) using rounding and dyn. prg.

   $\Rightarrow C_{max} \leq (1+\varepsilon) OPT$

2. Add short jobs ($\leq \varepsilon \cdot OPT$) to the schedule using LPT.

   $\Rightarrow C_{max} \leq (1+\varepsilon) OPT$