Parallel machine models: Makespan Minimization

Problem $P||C_{\text{max}}$:

- $m$ machines
- $n$ jobs with processing times $p_1, \ldots, p_n$
Parallel machine models: Makespan Minimization

Problem \( P||C_{\text{max}} \):

- \( m \) machines
- \( n \) jobs with processing times \( p_1, \ldots, p_n \)
- variable \( x_{ij} = \begin{cases} 1 & \text{if job } j \text{ is processed on machine } i \\ 0 & \text{else} \end{cases} \)
- ILP formulation:

\[
\begin{align*}
\min \quad & C_{\text{max}} \\
\text{s.t.} \quad & \sum_{j=1}^{n} x_{ij} p_j \leq C_{\text{max}} \quad i = 1, \ldots, m \\
& \sum_{i=1}^{m} x_{ij} = 1 \quad j = 1, \ldots, n \\
& x_{ij} \in \{0, 1\} \quad i = 1, \ldots, m; j = 1, \ldots, n
\end{align*}
\]
Parallel machine models: Makespan Minimization

Problem $P||C_{max}$:

- in lecture 2: $P2||C_{max}$ is NP-hard
- $P||C_{max}$ is even NP-hard in the strong sense (reduction from 3-PARTITION); i.e. also pseudopolynomial algorithms are unlikely
- question: What happens if $x_{ij} \in \{0, 1\}$ is relaxed?
Parallel machine models: Makespan Minimization

Problem $P\|C_{max}$:

- in lecture 2: $P2\|C_{max}$ is NP-hard
- $P\|C_{max}$ is even NP-hard in the strong sense (reduction from 3-PARTITION); i.e. also pseudopolynomial algorithms are unlikely
- question: What happens if $x_{ij} \in \{0,1\}$ is relaxed?
  answer: objective value of LP gets $\sum_{j=1}^{n} p_j / m$
- question: is this the optimal value of $P|pmtn|C_{max}$?
Parallel machine models: Makespan Minimization

Problem $P||C_{max}$:

- in lecture 2: $P2||C_{max}$ is NP-hard

- $P||C_{max}$ is even NP-hard in the strong sense (reduction from 3-PARTITION); i.e. also pseudopolynomial algorithms are unlikely

- question: What happens if $x_{ij} \in \{0, 1\}$ in the ILP is relaxed?
  answer: objective value of LP gets $\sum_{j=1}^{n} p_j / m$

- question: is this the optimal value of $P|pmtn|C_{max}$?
  answer: No!
  Example: $m = 2, n = 2, p = (1, 2)$
**Parallel machine models:** Makespan Minimization

**Problem \( P||C_{max} \):**

- in lecture 2: \( P2||C_{max} \) is NP-hard
- \( P||C_{max} \) is even NP-hard in the strong sense (reduction from 3-PARTITION); i.e. also pseudopolynomial algorithms are unlikely
- question: What happens if \( x_{ij} \in \{0, 1\} \) in the ILP is relaxed? 
  answer: objective value of LP gets \( \sum_{j=1}^{n} p_j / m \)
- question: is this the optimal value of \( P|pmtn|C_{max} \)?
  answer: No!
  Example: \( m = 2, n = 2, p = (1, 2) \)
- add \( C_{max} \geq p_j \) for \( j = 1, \ldots, m \) to ensure that each job has enough time
Parallel machine models: Makespan Minimization

LP for problem $P|\text{pmtn}|C_{\text{max}}$:

\[
\begin{align*}
\text{min} & \quad C_{\text{max}} \\
\text{s.t.} & \quad \sum_{j=1}^{n} x_{ij} p_j \leq C_{\text{max}} \quad i = 1, \ldots, m \\
& \quad p_j \leq C_{\text{max}} \quad j = 1, \ldots, n \\
& \quad \sum_{i=1}^{m} x_{ij} = 1 \quad j = 1, \ldots, n \\
& \quad x_{ij} \geq 0 \quad i = 1, \ldots, m; j = 1, \ldots, n
\end{align*}
\]
Parallel machine models: Makespan Minimization

LP for problem $P|p_{\text{mtm}}|C_{\text{max}}$:

$$\begin{align*}
\text{min} & \quad C_{\text{max}} \\
\text{s.t.} & \quad \sum_{j=1}^{n} x_{ij} p_j \leq C_{\text{max}} \quad i = 1, \ldots, m \\
& \quad p_j \leq C_{\text{max}} \quad j = 1, \ldots, n \\
& \quad \sum_{i=1}^{m} x_{ij} = 1 \quad j = 1, \ldots, n \\
& \quad x_{ij} \geq 0 \quad i = 1, \ldots, m; j = 1, \ldots, n
\end{align*}$$

- Optimal value of LP is $\max\{\max_{j=1}^{n} P_j, \sum_{j=1}^{n} p_j / m\}$
- LP gives no schedule, thus only a lower bound!
- construction of a schedule: simple (next slide) or via open shop (later)
Parallel machine models: Makespan Minimization

Wrap around rule for problem \( P|pmtn|C_{max} \):

- define \( opt := \max\{\max_{j=1}^{n} p_j, \sum_{j=1}^{n} p_j / m\} \)

- \( opt \) is a lower bound on the optimal value for problem \( P|pmtn|C_{max} \)

- Construction of a schedule with \( C_{max} = opt \):
  - fill the machines successively, schedule the jobs in any order and pre-empt a job if the time bound \( opt \) is met

- all jobs can be scheduled since \( opt \geq \sum_{j=1}^{n} p_j / m \)

- no job is scheduled at the same time on two machines since \( opt \geq \max_{j=1}^{n} p_j \)
**Parallel machine models**: Makespan Minimization

Wrap around rule for problem $P|pmtn|C_{max}$:

- Construction of a schedule with $C_{max} = opt$:
  fill the machines successively, schedule the jobs in any order and pre-empt a job if the time bound $opt$ is met

- all jobs can be scheduled since $opt \geq \sum_{j=1}^{n} p_j/m$

- no job is scheduled at the same time on two machines since $opt \geq \max_{j=1}^{n} p_j$

- Example: $m = 3, n = 5, p = (3, 7, 5, 1, 4)$

<table>
<thead>
<tr>
<th></th>
<th>M1</th>
<th>M2</th>
<th>M3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td>5</td>
</tr>
</tbody>
</table>

7
Parallel machine models: Makespan Minimization

Schedule construction via Open shop for $P|pmtn|C_{max}$:

- given an optimal solution $x$ of the LP, consider the following open shop instance
  
  - $n$ jobs, $m$ machines and $p_{ij} := x_{ij}p_j$

- solve for this instance $O|pmtn|C_{max}$
Parallel machine models: Makespan Minimization

Schedule construction via Open shop for $P|pmtn|C_{max}$:

- given an optimal solution $x$ of the LP, consider the open shop instance
  $n$ jobs, $m$ machines and $p_{ij} := x_{ij}p_j$

- solve for this instance $O|pmtn|C_{max}$

- Result: solution for problem $P|pmtn|C_{max}$

- for $O|pmtn|C_{max}$ we show later that an optimal solution has value

$$\max\{\max_{j=1}^{n} \sum_{i=1}^{m} p_{ij}, \max_{i=1}^{m} \sum_{j=1}^{n} p_{ij}\}$$

and can be calculated in polynomial time

- Result: solution of $O|pmtn|C_{max}$ is optimal for $P|pmtn|C_{max}$
Parallel machine models: Makespan Minimization

Uniform machines: $Q|pmtn|C_{max}$:

- $m$ machines with speeds $s_1, \ldots, s_m$
- $n$ jobs with processing times $p_1, \ldots, p_n$
- change LP!
Parallel machine models: Makespan Minimization

Uniform machines: $Q|pmtn|C_{max}$:

- $m$ machines with speeds $s_1, \ldots, s_m$
- $n$ jobs with processing times $p_1, \ldots, p_n$

$$\begin{align*}
\min \quad & C_{max} \\
\text{s.t.} \quad & \sum_{j=1}^{n} x_{ij} \frac{p_j}{s_i} \leq C_{max} \quad i = 1, \ldots, m \\
& \sum_{i=1}^{n} x_{ij} \frac{p_j}{s_i} \leq C_{max} \quad j = 1, \ldots, n \\
& \sum_{i=1}^{m} x_{ij} = 1 \quad j = 1, \ldots, n \\
& x_{ij} \geq 0 \quad i = 1, \ldots, m; j = 1, \ldots, n
\end{align*}$$
Parallel machine models: Makespan Minimization

Uniform machines: $Q|\text{pmtn}|C_{max}$ (cont.):

- since again no schedule is given, LP leads to lower bound for optimal value of $Q|\text{pmtn}|C_{max}$,

- as for $P|\text{pmtn}|C_{max}$ we may solve an open shop instance corresponding to the optimal solution $x$ of the LP with $n$ jobs, $m$ machines and $p_{ij} := x_{ij}p_j/s_i$

- this solution is an optimal schedule for $Q|\text{pmtn}|C_{max}$
Parallel machine models: Makespan Minimization

Unrelated machines: $R|pmtn|C_{max}$:

- $m$ machines
- $n$ jobs with processing times $p_1, \ldots, p_n$
- speed $s_{ij}$
- change LP!
Parallel machine models: Makespan Minimization

Unrelated machines: $R|pmtn|C_{max}$:

- $m$ machines
- $n$ jobs with processing times $p_1, \ldots, p_n$ and given speeds $s_{ij}$

\[
\begin{align*}
\text{min} & \quad C_{max} \\
\text{s.t.} & \quad \sum_{j=1}^{n} x_{ij} p_j / s_{ij} \leq C_{max} \quad i = 1, \ldots, m \\
& \quad \sum_{i=1}^{n} x_{ij} p_j / s_{ij} \leq C_{max} \quad j = 1, \ldots, n \\
& \quad \sum_{i=1}^{m} x_{ij} = 1 \quad j = 1, \ldots, n \\
& \quad x_{ij} \geq 0 \quad i = 1, \ldots, m; j = 1, \ldots, n
\end{align*}
\]
Parallel machine models: Makespan Minimization

Unrelated machines: $R|pmtn|C_{max}$ (cont.):

- same procedure as for $Q|pmtn|C_{max}$!
  - again no schedule is given,
  - LP leads to lower bound for optimal value of $R|pmtn|C_{max}$,
  - for optimal solution $x$ solve an corresponding open shop instance with $n$ jobs, $m$ machines and $p_{ij} := \frac{x_{ij} p_j}{s_{ij}}$
  - this solution is an optimal schedule for $R|pmtn|C_{max}$
Parallel machine models: Makespan Minimization

Approximation methods for: $P||C_{max}$:

- list scheduling methods (based on priority rules)
  - jobs are ordered in some sequence $\pi$
  - always when a machine gets free, the next unscheduled job in $\pi$ is assigned to that machine

- Theorem: List scheduling is a $(2 - 1/m)$-approximation for problem $P||C_{max}$ for any given sequence $\pi$

- Proof on the board

- Holds also for $P|r_j|C_{max}$
Parallel machine models: Makespan Minimization

Approximation methods for: $P||C_{max}$ (cont.):

- consider special list
- LPT-rule (longest processing time first) is a natural candidate
- **Theorem**: The LPT-rule leads to a $(4/3 - 1/3m)$-approximation for problem $P||C_{max}$
  - Proof on the board uses following result:
  - **Lemma**: If an optimal schedule for problem $P||C_{max}$ results in at most 2 jobs on any machine, then the LPT-rule is optimal
  - Proof as Exercise
- the bound $(4/3 - 1/3m)$ is tied (Exercise)
Parallel machine models: Total Completion Time

Parallel machines: $P|\sum C_j$:

- for $m = 1$, the SPT-rule is optimal (see Lecture 2)
- for $m \geq 2$ a partition of the jobs is needed
- if a job $j$ is scheduled as $k$-last job on a machine, this job contributes $k p_j$ to the objective value
Parallel machine models: Total Completion Time

Parallel machines: $P||\sum C_j$:

- for $m = 1$, the SPT-rule is optimal (see Lecture 2)
- for $m \geq 2$ a partition of the jobs is needed
- if a job $j$ is scheduled as $k$-last job on a machine, this job contributes $k p_j$ to the objective value
- we have $m$ last positions where the processing time is weighted by 1, $m$ second last positions where the processing time is weighted by 2, etc.
- use the $n$ smallest weights for positioning the jobs
**Parallel machine models:** Total Completion Time

Parallel machines: $P|| \sum C_j$:

- for $m = 1$, the SPT-rule is optimal (see Lecture 2)
- for $m \geq 2$ a partition of the jobs is needed
- if a job $j$ is scheduled as $k$-last job on a machine, this job contributes $kp_j$ to the objective value
- we have $m$ last positions where the processing time is weighted by 1, $m$ second last positions where the processing time is weighted by 2, etc.
- use the $n$ smallest weights for positioning the jobs
- assign job with the $i$th largest processing time to $i$th smallest weight
- **Result:** SPT is also optimal for $P|| \sum C_j$
Parallel machine models: Total Completion Time

Uniform machines: $Q||\sum C_j$:

- if a job $j$ is scheduled as $k$-last job on a machine $M_r$, this job contributes $kp_j/s_r = (k/s_r)p_j$ to the objective value; i.e. job $j$ gets ’weight’ $(k/s_r)$

- for scheduling the $n$ jobs on the $m$ machines, we have weights

$$\left\{ \frac{1}{s_1}, \ldots, \frac{1}{s_m}, \frac{2}{s_1}, \ldots, \frac{2}{s_m}, \ldots, \frac{n}{s_1}, \ldots, \frac{n}{s_m} \right\}$$

- from these $nm$ weights we select the $n$ smallest weights and assign the $i$th largest job to the $i$th smallest weight leading to an optimal schedule
Parallel machine models: Total Completion Time

Example uniform machines: $Q||\sum C_j$:

- $n = 6$, $p = (6, 9, 8, 12, 4, 2)$
- $m = 3$, $s = (3, 1, 4)$
- possible weights:
  $$\left\{3', 1', 4', 3', 1', 4' \right\}$$
- 6 smallest weights:
  $$\left\{3', 1', 4', 3', 1', 4' \right\}$$
Parallel machine models: Total Completion Time

Example uniform machines: $Q\|\sum C_j$:

- $n = 6$, $p = (6, 9, 8, 12, 4, 2)$
- $m = 3$, $s = (3, 1, 4)$
- 6 smallest weights:
  \[
  \left\{ \frac{1}{3}, \frac{1}{4}, \frac{1}{3}, \frac{2}{4}, \frac{2}{3}, \frac{3}{4}, \frac{3}{3}, \frac{3}{4}, \frac{4}{4}, \frac{4}{4}, \frac{5}{5}, \frac{5}{5}, \frac{6}{6}, \frac{6}{6}, \frac{6}{6} \right\}
  \]
- sorted list of weights:
  \[
  \left\{ \frac{1}{4}, \frac{1}{3}, \frac{2}{3}, \frac{2}{4}, \frac{3}{4}, \frac{3}{3}, \frac{4}{4} \right\}
  \]
- jobs sorted by decreasing processing times: $(4, 2, 3, 1, 5, 6)$
**Parallel machine models:** Total Completion Time

**Example uniform machines:** $Q||\sum C_j$:

- $n = 6, \ p = (6, 9, 8, 12, 4, 2)$
- $m = 3, \ s = (3, 1, 4)$
- sorted list of weights:
  \[
  \{ \frac{1}{4}, \frac{1}{3}, \frac{2}{4}, \frac{2}{3}, \frac{3}{4} \}
  \]
- jobs sorted by decreasing processing times: $(4, 2, 3, 1, 5, 6)$

- Schedule:

<table>
<thead>
<tr>
<th></th>
<th>M1</th>
<th>M2</th>
<th>M3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>
Parallel machine models: Total Completion Time

Unrelated machines: $R||\sum C_j$:

- if a job $j$ is scheduled as $k$-last job on a machine $M_r$, this job contributes $k p_{rj}$ to the objective value;
- since now the 'weight' is also job-dependent, we cannot simply sort the 'weights';
- assignment problem:
  - $n$ jobs
  - $nm$ machine positions $(k, r)$ ($k$-last position on $M_r$)
  - assigning job $j$ to $(k, r)$ has costs $k p_{rj}$
  - find an assignment of minimal costs of all jobs to machine positions
- leads to optimal solution of $R||\sum C_j$ in polynomial time
Parallel machine models: Total Weighted Completion Time

Parallel machines: $P|| \sum w_j C_j$:

- Problem 1$|| \sum w_j C_j$ is solvable via the WSPT-rule (Lecture 2)
- Problem $P2|| \sum w_j C_j$ is . . .
Parallel machine models: Total Weighted Completion Time

Parallel machines: $P||\sum w_jC_j$:

- Problem $1||\sum w_jC_j$ is solvable via the WSPT-rule (Lecture 2)
- Problem $P2||\sum w_jC_j$ is already NP-hard, but
- Problem $P2||\sum w_jC_j$ is pseudopolynomial solvable
- Problem $P||\sum w_jC_j$ is NP-hard in the strong sense
  - Proof by reduction using 3-PARTITION as exercise
- Approximation:
Parallel machine models: Total Weighted Completion Time

Parallel machines: $P||\sum w_j C_j$:

- **Problem $1||\sum w_j C_j$** is solvable via the WSPT-rule (Lecture 2)
- **Problem $P2||\sum w_j C_j$** is already NP-hard, but
- **Problem $P2||\sum w_j C_j$** is pseudopolynomial solvable
- **Problem $P||\sum w_j C_j$** is NP-hard in the strong sense
  Proof by reduction using 3-PARTITION as exercise

**Approximation**: the WSPT-rule gives an $\frac{1}{2}(1 + \sqrt{2})$ approximation
Proof is not given; uses fact that worst case examples have equal $w_j/p_j$ ratios for all jobs