# Approximation Algorithms

## Lecture 7:
## Scheduling Jobs on Parallel Machines

### Part I:
### ILP & Parametric Pruning

Alexander Wolff

Winter 2022/23

# Scheduling on Parallel Machines

**Given:** A set $\mathcal{J}$ of **jobs**,

$$\mathcal{J} = \{J_1, J_2, \ldots, J_8\}$$

# Scheduling on Parallel Machines

**Given:** A set $\mathcal{J}$ of **jobs**,
a set $\mathcal{M}$ of **machines**, and

$$\mathcal{J} = \{J_1, J_2, \dots, J_8\}$$

$$\mathcal{M} = \{M_1, M_2, M_3\}$$

# Scheduling on Parallel Machines

**Given:** A set $\mathcal{J}$ of **jobs**,
a set $\mathcal{M}$ of **machines**, and
for each $M_i \in \mathcal{M}$ and $J_j \in \mathcal{J}$
the **processing time** $p_{ij} \in \mathbb{N}^+$ of $J_j$ on $M_i$.

$$\mathcal{J} = \{J_1, J_2, \dots, J_8\}$$

$$\mathcal{M} = \{M_1, M_2, M_3\}$$

$$(p_{ij})_{M_i \in \mathcal{M}, J_j \in \mathcal{J}}$$

# Scheduling on Parallel Machines

**Given:** A set $\mathcal{J}$ of **jobs**,
a set $\mathcal{M}$ of **machines**, and
for each $M_i \in \mathcal{M}$ and $J_j \in \mathcal{J}$
the **processing time** $p_{ij} \in \mathbb{N}^+$ of $J_j$ on $M_i$.

**Task:** A **schedule** $\sigma\colon \mathcal{J} \to \mathcal{M}$ of the jobs on the machines that minimizes the total time to completion (**makespan**), i.e., minimizes the maximum time a machine is in use.

$$\mathcal{J} = \{J_1, J_2, \dots, J_8\}$$

$$\mathcal{M} = \{M_1, M_2, M_3\}$$

$$(p_{ij})_{M_i \in \mathcal{M}, J_j \in \mathcal{J}}$$

# Scheduling on Parallel Machines

**Given:** A set $\mathcal{J}$ of **jobs**,
a set $\mathcal{M}$ of **machines**, and
for each $M_i \in \mathcal{M}$ and $J_j \in \mathcal{J}$
the **processing time** $p_{ij} \in \mathbb{N}^+$ of $J_j$ on $M_i$.

**Task:** A **schedule** $\sigma\colon \mathcal{J} \to \mathcal{M}$ of the jobs on the machines that minimizes the total time to completion (**makespan**), i.e., minimizes the maximum time a machine is in use.

$M_1$

$M_2$

$M_3$

$$\mathcal{J} = \{J_1, J_2, \dots, J_8\}$$

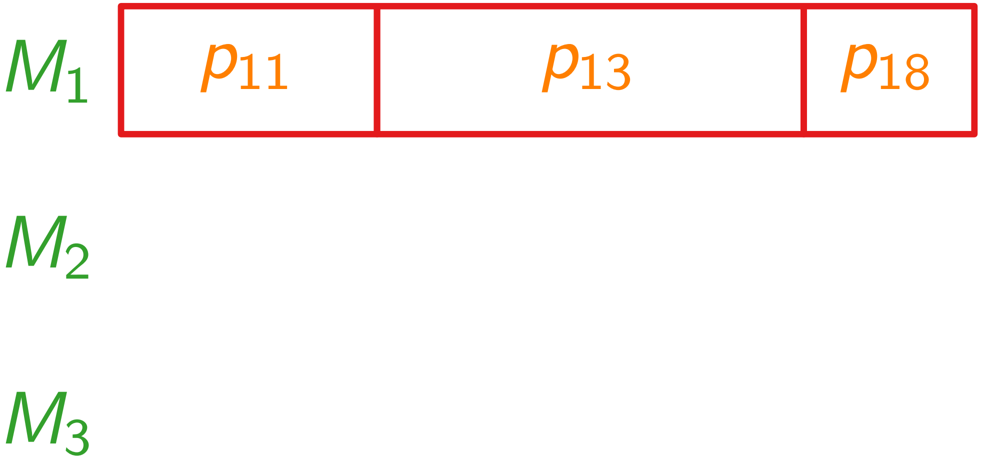
$$\mathcal{M} = \{M_1, M_2, M_3\}$$

$$(p_{ij})_{M_i \in \mathcal{M}, J_j \in \mathcal{J}}$$

# Scheduling on Parallel Machines

**Given:** A set $\mathcal{J}$ of **jobs**,
a set $\mathcal{M}$ of **machines**, and
for each $M_i \in \mathcal{M}$ and $J_j \in \mathcal{J}$
the **processing time** $p_{ij} \in \mathbb{N}^+$ of $J_j$ on $M_i$.

**Task:** A **schedule** $\sigma\colon \mathcal{J} \to \mathcal{M}$ of the jobs on the machines that minimizes the total time to completion (**makespan**), i.e., minimizes the maximum time a machine is in use.



$$\mathcal{J} = \{J_1, J_2, \dots, J_8\}$$

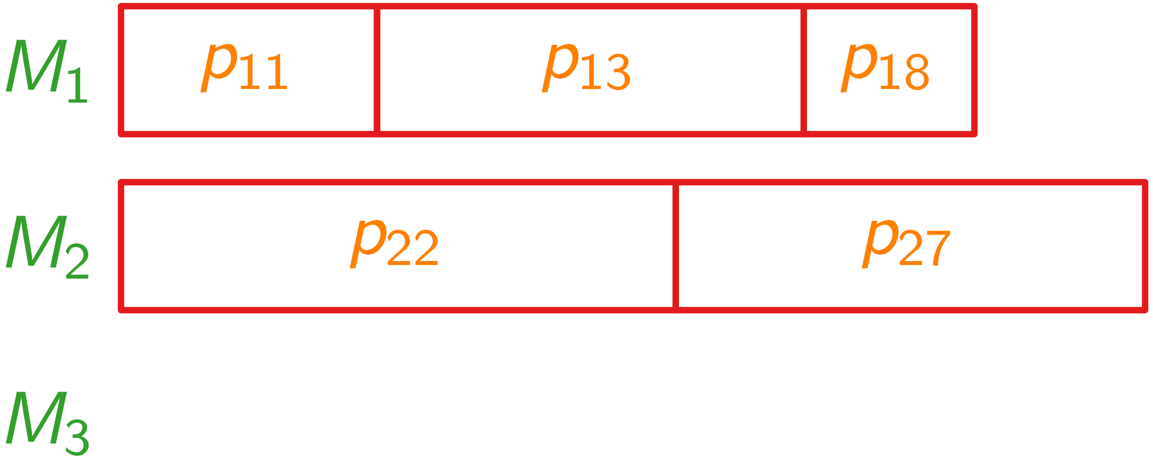
$$\mathcal{M} = \{M_1, M_2, M_3\}$$

$$(p_{ij})_{M_i \in \mathcal{M}, J_j \in \mathcal{J}}$$

# Scheduling on Parallel Machines

**Given:** A set $\mathcal{J}$ of **jobs**,
a set $\mathcal{M}$ of **machines**, and
for each $M_i \in \mathcal{M}$ and $J_j \in \mathcal{J}$
the **processing time** $p_{ij} \in \mathbb{N}^+$ of $J_j$ on $M_i$.

**Task:** A **schedule** $\sigma \colon \mathcal{J} \to \mathcal{M}$ of the jobs on the machines that minimizes the total time to completion (**makespan**), i.e., minimizes the maximum time a machine is in use.



$\mathcal{J} = \{J_1, J_2, \dots, J_8\}$

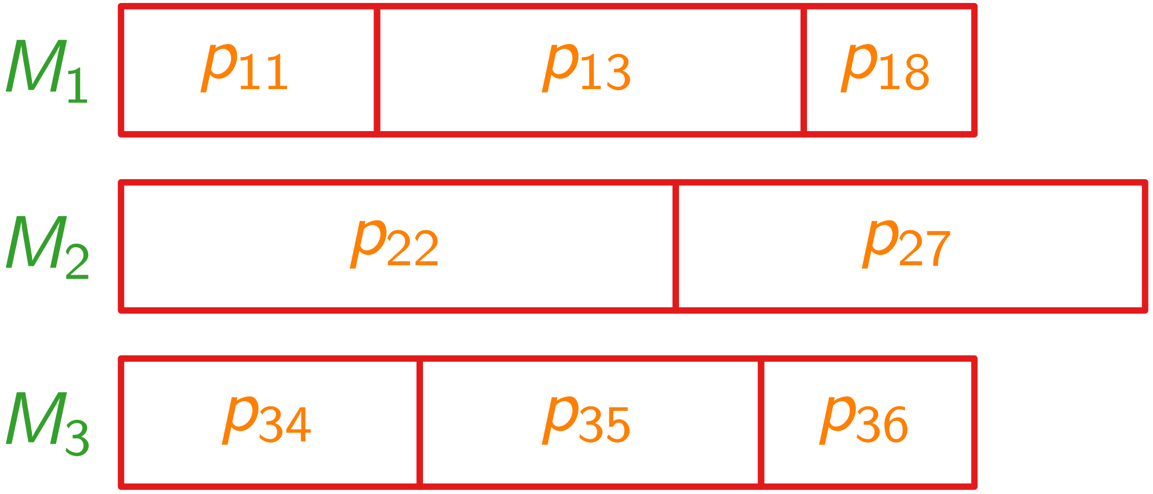
$\mathcal{M} = \{M_1, M_2, M_3\}$

$(p_{ij})_{M_i \in \mathcal{M}, J_j \in \mathcal{J}}$

# Scheduling on Parallel Machines

**Given:** A set $\mathcal{J}$ of **jobs**,
a set $\mathcal{M}$ of **machines**, and
for each $M_i \in \mathcal{M}$ and $J_j \in \mathcal{J}$
the **processing time** $p_{ij} \in \mathbb{N}^+$ of $J_j$ on $M_i$.

**Task:** A **schedule** $\sigma\colon \mathcal{J} \to \mathcal{M}$ of the jobs on the machines that minimizes the total time to completion (**makespan**), i.e., minimizes the maximum time a machine is in use.


$M_1$: $p_{11}$, $p_{13}$, $p_{18}$

$M_2$: $p_{22}$, $p_{27}$

$M_3$: $p_{34}$, $p_{35}$, $p_{36}$

$$\mathcal{J} = \{J_1, J_2, \ldots, J_8\}$$

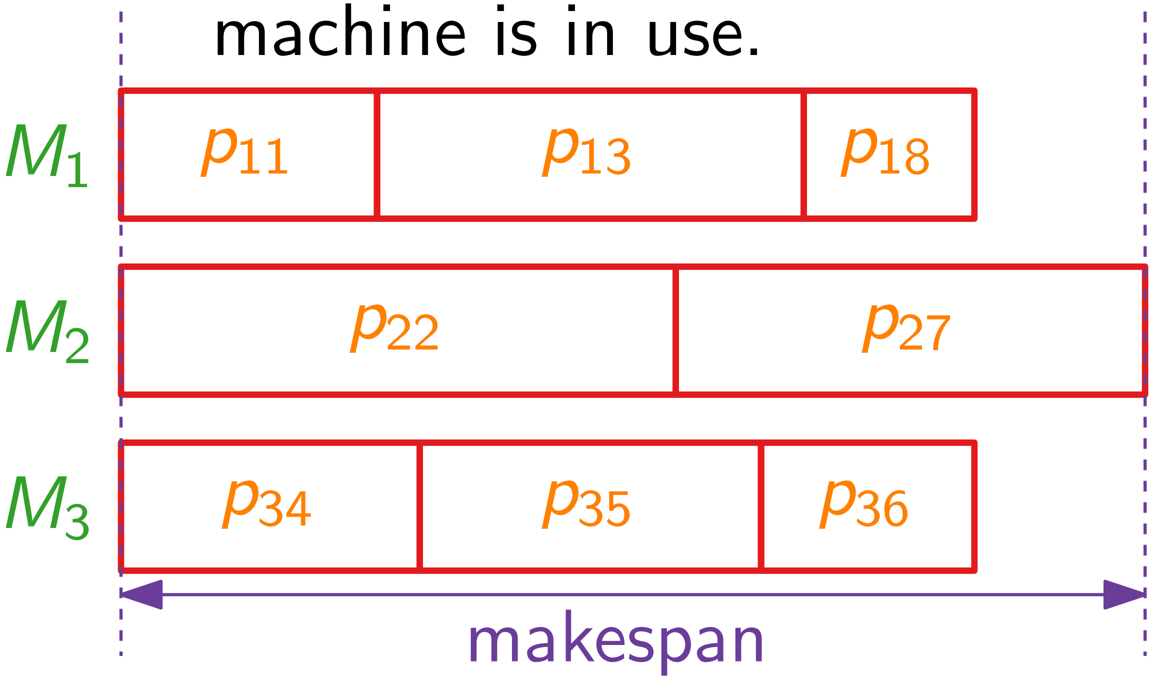
$$\mathcal{M} = \{M_1, M_2, M_3\}$$

$$(p_{ij})_{M_i \in \mathcal{M}, J_j \in \mathcal{J}}$$

# Scheduling on Parallel Machines

**Given:** A set $\mathcal{J}$ of **jobs**,
a set $\mathcal{M}$ of **machines**, and
for each $M_i \in \mathcal{M}$ and $J_j \in \mathcal{J}$
the **processing time** $p_{ij} \in \mathbb{N}^+$ of $J_j$ on $M_i$.

**Task:** A **schedule** $\sigma \colon \mathcal{J} \to \mathcal{M}$ of the jobs on the machines that minimizes the total time to completion (**makespan**), i.e., minimizes the maximum time a machine is in use.



$\mathcal{J} = \{J_1, J_2, \dots, J_8\}$

$\mathcal{M} = \{M_1, M_2, M_3\}$

$(p_{ij})_{M_i \in \mathcal{M}, J_j \in \mathcal{J}}$

# Formulation as ILP

**minimize** $t$

**subject to**

# Formulation as ILP

**minimize** $t$

**subject to**

$$x_{ij} \in \{0,1\}, \qquad M_i \in \mathcal{M}, J_j \in \mathcal{J}$$

# Formulation as ILP

$$
\begin{array}{lll}
\textbf{minimize} & t & \\
\textbf{subject to} & & J_j \in \mathcal{J} \\
 & & M_i \in \mathcal{M} \\
 & x_{ij} \in \{0,1\}, & M_i \in \mathcal{M}, J_j \in \mathcal{J}
\end{array}
$$

# Formulation as ILP

$$\begin{array}{lll} \textbf{minimize} & t & \\ \textbf{subject to} & \displaystyle\sum_{M_i \in \mathcal{M}} x_{ij} = 1, & J_j \in \mathcal{J} \\ & & M_i \in \mathcal{M} \\ & x_{ij} \in \{0,1\}, & M_i \in \mathcal{M}, J_j \in \mathcal{J} \end{array}$$

# Formulation as ILP

$$\begin{aligned}
\textbf{minimize} \quad & t \\
\textbf{subject to} \quad & \sum_{M_i \in \mathcal{M}} x_{ij} = 1, && J_j \in \mathcal{J} \\
& \sum_{J_j \in \mathcal{J}} x_{ij} p_{ij} \leq t, && M_i \in \mathcal{M} \\
& x_{ij} \in \{0, 1\}, && M_i \in \mathcal{M}, J_j \in \mathcal{J}
\end{aligned}$$

# Formulation as ILP

$$\begin{array}{lll}
\textbf{minimize} & t & \\
\textbf{subject to} & \displaystyle\sum_{M_i \in \mathcal{M}} x_{ij} = 1, & J_j \in \mathcal{J} \\
& \displaystyle\sum_{J_j \in \mathcal{J}} x_{ij} p_{ij} \leq t, & M_i \in \mathcal{M} \\
& x_{ij} \in \{0,1\}, & M_i \in \mathcal{M}, J_j \in \mathcal{J}
\end{array}$$

**Task:** Prove that the integrality gap is unbounded!

# Formulation as ILP

$$\begin{array}{lll}
\textbf{minimize} & t & \\
\textbf{subject to} & \displaystyle\sum_{M_i \in \mathcal{M}} x_{ij} = 1, & J_j \in \mathcal{J} \\
& \displaystyle\sum_{J_j \in \mathcal{J}} x_{ij} p_{ij} \leq t, & M_i \in \mathcal{M} \\
& x_{ij} \in \{0, 1\}, & M_i \in \mathcal{M}, J_j \in \mathcal{J}
\end{array}$$

**Task:** Prove that the integrality gap is unbounded!

**Solution:** $m$ machines and one job with processing time $m$

# Formulation as ILP

$$
\begin{array}{lll}
\textbf{minimize} & t & \\
\textbf{subject to} & \displaystyle\sum_{M_i \in \mathcal{M}} x_{ij} = 1, & J_j \in \mathcal{J} \\
 & \displaystyle\sum_{J_j \in \mathcal{J}} x_{ij} p_{ij} \leq t, & M_i \in \mathcal{M} \\
 & x_{ij} \in \{0, 1\}, & M_i \in \mathcal{M}, J_j \in \mathcal{J}
\end{array}
$$

**Task:** Prove that the integrality gap is unbounded!

**Solution:** $m$ machines and one job with processing time $m$
$\Rightarrow \mathrm{OPT} = m$ and $\mathrm{OPT}_{\mathrm{frac}} = 1$.

# Parametric Pruning

Strengthen the ILP → implicit (non-linear) constraint:

If $p_{ij} > t$, then set $x_{ij} = 0$.

# Parametric Pruning

Strengthen the ILP → implicit (non-linear) constraint:

If $p_{ij} > t$, then set $x_{ij} = 0$.

Introduce new parameter $T \in \mathbb{N}$ as a lower bound on OPT.

# Parametric Pruning

Strengthen the ILP → implicit (non-linear) constraint:

If $p_{ij} > t$, then set $x_{ij} = 0$.

Introduce new parameter $T \in \mathbb{N}$ as a lower bound on OPT.

Define $S_T := \{\, (i, j) \colon M_i \in \mathcal{M},\ J_j \in \mathcal{J},\ p_{ij} \leq T \,\}$.

# Parametric Pruning

Strengthen the ILP → implicit (non-linear) constraint:

If $p_{ij} > t$, then set $x_{ij} = 0$.

Introduce new parameter $T \in \mathbb{N}$ as a lower bound on OPT.

Define $S_T := \{\, (i, j) \colon M_i \in \mathcal{M},\ J_j \in \mathcal{J},\ p_{ij} \leq T \,\}$.

Define the “pruned” relaxation LP($T$):

**minimize** $t$

**subject to**

$$\sum_{M_i \in \mathcal{M}} x_{ij} = 1, \qquad J_j \in \mathcal{J}$$

$$\sum_{J_j \in \mathcal{J}} x_{ij} p_{ij} \leq t, \qquad M_i \in \mathcal{M}$$

$$x_{ij} \in \{0, 1\}, \qquad M_i \in \mathcal{M}, J_j \in \mathcal{J}$$

# Parametric Pruning

Strengthen the ILP → implicit (non-linear) constraint:
If $p_{ij} > t$, then set $x_{ij} = 0$.

Introduce new parameter $T \in \mathbb{N}$ as a lower bound on OPT.

Define $S_T := \{\, (i,j) \colon M_i \in \mathcal{M},\ J_j \in \mathcal{J},\ p_{ij} \leq T \,\}$.

Define the "pruned" relaxation LP($T$):

$$
\begin{array}{lll}
\textbf{minimize} & t & \\
\textbf{subject to} & \displaystyle\sum_{M_i \in \mathcal{M}} x_{ij} = 1, & J_j \in \mathcal{J} \\
& \displaystyle\sum_{J_j \in \mathcal{J}} x_{ij} p_{ij} \leq t, & M_i \in \mathcal{M} \\
& x_{ij} \cancel{\in \{0,1\},} \geq 0 & M_i \in \mathcal{M},\ J_j \in \mathcal{J}
\end{array}
$$

# Parametric Pruning

Strengthen the ILP → implicit (non-linear) constraint:

If $p_{ij} > t$, then set $x_{ij} = 0$.

Introduce new parameter $T \in \mathbb{N}$ as a lower bound on OPT.

Define $S_T := \{\, (i,j) \colon M_i \in \mathcal{M},\ J_j \in \mathcal{J},\ p_{ij} \leq T \,\}$.

Define the “pruned” relaxation LP($T$):

$$\begin{array}{lll}
\textbf{minimize} & t & \\
\textbf{subject to} & \displaystyle\sum_{M_i \in \mathcal{M}} x_{ij} = 1, & J_j \in \mathcal{J} \\
 & \displaystyle\sum_{J_j \in \mathcal{J}} x_{ij} p_{ij} \leq t, & M_i \in \mathcal{M} \\
 & x_{ij} \cancel{\in \{0,1\},} \geq 0 & \cancel{M_i \in \mathcal{M}, J_j \in \mathcal{J}} \quad (i,j) \in S_T
\end{array}$$

# Parametric Pruning

Strengthen the ILP → implicit (non-linear) constraint:

If $p_{ij} > t$, then set $x_{ij} = 0$.

Introduce new parameter $T \in \mathbb{N}$ as a lower bound on OPT.

Define $S_T := \{\, (i,j) \colon M_i \in \mathcal{M},\ J_j \in \mathcal{J},\ p_{ij} \le T \,\}$.

Define the “pruned” relaxation LP($T$):

$$\begin{array}{lll}
\textbf{minimize} & t & \\
\textbf{subject to} & \displaystyle\sum_{i\colon (i,j)\in S_T\, \xcancel{M_i\in\mathcal{M}}} x_{ij} = 1, & J_j \in \mathcal{J} \\
 & \displaystyle\sum_{J_j\in\mathcal{J}} x_{ij}\,p_{ij} \le t, & M_i \in \mathcal{M} \\
 & x_{ij}\, \xcancel{\in \{0,1\}}, \ge 0 \quad \xcancel{M_i \in \mathcal{M},\ J_j \in \mathcal{J}} & (i,j) \in S_T
\end{array}$$

# Parametric Pruning

Strengthen the ILP → implicit (non-linear) constraint:

If $p_{ij} > t$, then set $x_{ij} = 0$.

Introduce new parameter $T \in \mathbb{N}$ as a lower bound on OPT.

Define $S_T := \{\, (i,j) \colon M_i \in \mathcal{M},\ J_j \in \mathcal{J},\ p_{ij} \le T \,\}$.

Define the “pruned” relaxation LP($T$):

$$\begin{array}{lll}
\textbf{minimize} & t & \\
\textbf{subject to} & \displaystyle\sum_{i\colon (i,j)\in S_T\, \cancel{M_i\in\mathcal{M}}} x_{ij} = 1, & J_j \in \mathcal{J} \\
& \displaystyle\sum_{j\colon (i,j)\in S_T\, \cancel{J_j\in\mathcal{J}}} x_{ij}\, p_{ij} \le t, & M_i \in \mathcal{M} \\
& x_{ij} \cancel{\in \{0,1\}}, \ge 0 \quad \cancel{M_i \in \mathcal{M}, J_j \in \mathcal{J}} & (i,j) \in S_T
\end{array}$$

# Parametric Pruning

Strengthen the ILP → implicit (non-linear) constraint:

If $p_{ij} > t$, then set $x_{ij} = 0$.

Introduce new parameter $T \in \mathbb{N}$ as a lower bound on OPT.

Define $S_T := \{\, (i,j) \colon M_i \in \mathcal{M},\ J_j \in \mathcal{J},\ p_{ij} \leq T \,\}$.

Define the "pruned" relaxation LP($T$):

$$
\begin{array}{lll}
\textbf{minimize} & t & \\
\textbf{subject to} & \displaystyle\sum_{i\colon (i,j) \in S_T \cancel{M_i \in \mathcal{M}}} x_{ij} = 1, & J_j \in \mathcal{J} \\
 & \displaystyle\sum_{j\colon (i,j) \in S_T \cancel{J_j \in \mathcal{J}}} x_{ij} p_{ij} \leq \cancel{t,}\, T & M_i \in \mathcal{M} \\
 & x_{ij} \cancel{\in \{0,1\},} \geq 0 \ \cancel{M_i \in \mathcal{M}, J_j \in \mathcal{J}} & (i,j) \in S_T
\end{array}
$$

# Parametric Pruning

Strengthen the ILP → implicit (non-linear) constraint:

If $p_{ij} > t$, then set $x_{ij} = 0$.

Introduce new parameter $T \in \mathbb{N}$ as a lower bound on OPT.

Define $S_T := \{\, (i,j) \colon M_i \in \mathcal{M},\ J_j \in \mathcal{J},\ p_{ij} \le T \,\}$.

Define the “pruned” relaxation LP($T$):

$$\cancel{\textbf{minimize}\quad t}$$

$$\textbf{subject to}\quad \sum_{i\colon (i,j)\in S_T\, \cancel{M_i\in\mathcal{M}}} x_{ij} = 1, \qquad J_j \in \mathcal{J}$$

$$\sum_{j\colon (i,j)\in S_T\, \cancel{J_j\in\mathcal{J}}} x_{ij}p_{ij} \le \cancel{t,}\ T \quad M_i \in \mathcal{M}$$

$$x_{ij} \cancel{\in \{0,1\},} \ge 0 \quad \cancel{M_i \in \mathcal{M},\ J_j \in \mathcal{J}} \quad (i,j) \in S_T$$

# Parametric Pruning

Strengthen the ILP → implicit (non-linear) constraint:

If $p_{ij} > t$, then set $x_{ij} = 0$.

Introduce new parameter $T \in \mathbb{N}$ as a lower bound on OPT.

Define $S_T := \{\, (i,j) \colon M_i \in \mathcal{M},\ J_j \in \mathcal{J},\ p_{ij} \leq T \,\}$.

Define the “pruned” relaxation LP($T$):

$$\boxed{\begin{aligned} \sum_{i\colon (i,j) \in S_T} x_{ij} &= 1, && J_j \in \mathcal{J} \\ \sum_{j\colon (i,j) \in S_T} x_{ij} p_{ij} &\leq T, && M_i \in \mathcal{M} \\ x_{ij} &\geq 0, && (i,j) \in S_T \end{aligned}}$$

# Parametric Pruning

Strengthen the ILP → implicit (non-linear) constraint:

If $p_{ij} > t$, then set $x_{ij} = 0$.

Introduce new parameter $T \in \mathbb{N}$ as a lower bound on OPT.

Define $S_T := \{\, (i,j) \colon M_i \in \mathcal{M},\ J_j \in \mathcal{J},\ p_{ij} \leq T \,\}$.

Define the “pruned” relaxation LP($T$):

$$\boxed{\begin{aligned} \sum_{i\colon (i,j)\in S_T} x_{ij} &= 1, & J_j &\in \mathcal{J} \\ \sum_{j\colon (i,j)\in S_T} x_{ij} p_{ij} &\leq T, & M_i &\in \mathcal{M} \\ x_{ij} &\geq 0, & (i,j) &\in S_T \end{aligned}}$$

**Note:**

LP($T$) has no objective function; we just need to check whether a feasible solution exists.

# Parametric Pruning

Strengthen the ILP → implicit (non-linear) constraint:

If $p_{ij} > t$, then set $x_{ij} = 0$.

Introduce new parameter $T \in \mathbb{N}$ as a lower bound on OPT.

Define $S_T := \{\, (i,j) \colon M_i \in \mathcal{M},\, J_j \in \mathcal{J},\, p_{ij} \le T \,\}$.

Define the “pruned” relaxation LP($T$):

$$\boxed{\begin{aligned} \sum_{i\colon (i,j)\in S_T} x_{ij} &= 1, && J_j \in \mathcal{J} \\ \sum_{j\colon (i,j)\in S_T} x_{ij} p_{ij} &\le T, && M_i \in \mathcal{M} \\ x_{ij} &\ge 0, && (i,j) \in S_T \end{aligned}}$$

**Note:**

LP($T$) has no objective function; we just need to check whether a feasible solution exists.

But why does this LP give a good integrality gap?

# Approximation Algorithms

## Lecture 7:
## Scheduling Jobs on Parallel Machines

### Part II:
### Properties of Extreme-Point Solutions

# Properties of Extreme Point Solutions

Use binary search to find the smallest $T$ so that LP($T$) has a solution.

LP($T$):

$$\boxed{\begin{aligned} \sum_{i\colon (i,j)\in S_T} x_{ij} &= 1, \quad && J_j \in \mathcal{J} \\ \sum_{j\colon (i,j)\in S_T} x_{ij} p_{ij} &\leq T, \quad && M_i \in \mathcal{M} \\ x_{ij} &\geq 0, \quad && (i,j) \in S_T \end{aligned}}$$

# Properties of Extreme Point Solutions

Use binary search to find the smallest $T$ so that LP($T$) has a solution. Let $T^*$ be this value of $T$.

LP($T$):

$$\boxed{\begin{aligned} \sum_{i:\,(i,j)\in S_T} x_{ij} &= 1, && J_j \in \mathcal{J} \\ \sum_{j:\,(i,j)\in S_T} x_{ij} p_{ij} &\leq T, && M_i \in \mathcal{M} \\ x_{ij} &\geq 0, && (i,j) \in S_T \end{aligned}}$$

# Properties of Extreme Point Solutions

Use binary search to find the smallest $T$ so that LP($T$) has a solution. Let $T^*$ be this value of $T$.

What are the bounds for our search?

LP($T$):

$$\boxed{\begin{aligned} \sum_{i:\,(i,j)\in S_T} x_{ij} &= 1, && J_j \in \mathcal{J} \\ \sum_{j:\,(i,j)\in S_T} x_{ij} p_{ij} &\leq T, && M_i \in \mathcal{M} \\ x_{ij} &\geq 0, && (i,j) \in S_T \end{aligned}}$$

# Properties of Extreme Point Solutions

Use binary search to find the smallest $T$ so that LP($T$) has a solution. Let $T^*$ be this value of $T$.

What are the bounds for our search?

**Observe:** $T^* \leq \mathrm{OPT}$

LP($T$):

$$\begin{aligned} \sum_{i\,:\,(i,j)\in S_T} x_{ij} &= 1, && J_j \in \mathcal{J} \\ \sum_{j\,:\,(i,j)\in S_T} x_{ij} p_{ij} &\leq T, && M_i \in \mathcal{M} \\ x_{ij} &\geq 0, && (i,j) \in S_T \end{aligned}$$

# Properties of Extreme Point Solutions

Use binary search to find the smallest $T$ so that LP($T$) has a solution. Let $T^*$ be this value of $T$.

What are the bounds for our search?

**Observe:** $T^* \leq \text{OPT}$

**Idea:** Round an extreme-point solution of LP($T^*$) to a schedule whose makespan is at most $2T^*$.

LP($T$):

$$\begin{aligned} \sum_{i:\,(i,j)\in S_T} x_{ij} &= 1, && J_j \in \mathcal{J} \\ \sum_{j:\,(i,j)\in S_T} x_{ij} p_{ij} &\leq T, && M_i \in \mathcal{M} \\ x_{ij} &\geq 0, && (i,j) \in S_T \end{aligned}$$

# Properties of Extreme Point Solutions

Use binary search to find the smallest $T$ so that LP($T$) has a solution. Let $T^*$ be this value of $T$.

What are the bounds for our search?

**Observe:** $T^* \leq \text{OPT}$

**Idea:** Round an extreme-point solution of LP($T^*$) to a schedule whose makespan is at most $2T^*$.

LP($T$):

$$\begin{aligned} \sum_{i\colon (i,j) \in S_T} x_{ij} &= 1, && J_j \in \mathcal{J} \\ \sum_{j\colon (i,j) \in S_T} x_{ij} p_{ij} &\leq T, && M_i \in \mathcal{M} \\ x_{ij} &\geq 0, && (i,j) \in S_T \end{aligned}$$

**Lemma 1.**
Every extreme-point solution of LP($T$) has at most $|\mathcal{M}| + |\mathcal{J}|$ positive variables.

# Properties of Extreme Point Solutions

Use binary search to find the smallest $T$ so that LP($T$) has a solution. Let $T^*$ be this value of $T$.

What are the bounds for our search?

**Observe:** $T^* \leq \mathrm{OPT}$

**Idea:** Round an extreme-point solution of LP($T^*$) to a schedule whose makespan is at most $2T^*$.

LP($T$):

$$\begin{aligned} \sum_{i\colon (i,j)\in S_T} x_{ij} &= 1, & J_j &\in \mathcal{J} \\ \sum_{j\colon (i,j)\in S_T} x_{ij} p_{ij} &\leq T, & M_i &\in \mathcal{M} \\ x_{ij} &\geq 0, & (i,j) &\in S_T \end{aligned}$$

**Lemma 1.**
Every extreme-point solution of LP($T$) has at most $|\mathcal{M}| + |\mathcal{J}|$ positive variables.

**Lemma 2.**
Every extreme-point solution of LP($T$) sets at least $|\mathcal{J}| - |\mathcal{M}|$ jobs integrally.

# Lemma 1

$$\begin{aligned}
\sum_{i:\,(i,j)\in S_T} x_{ij} &= 1, && J_j \in \mathcal{J}\\
\sum_{j:\,(i,j)\in S_T} x_{ij} p_{ij} &\le T, && M_i \in \mathcal{M}\\
x_{ij} &\ge 0, && (i,j) \in S_T
\end{aligned}$$

**Lemma 1.**
Every extreme-point solution of LP($T$) has at most $|\mathcal{M}| + |\mathcal{J}|$ positive variables.

# Lemma 1

$$
\begin{aligned}
\sum_{i\colon (i,j)\in S_T} x_{ij} &= 1, && J_j \in \mathcal{J}\\
\sum_{j\colon (i,j)\in S_T} x_{ij} p_{ij} &\le T, && M_i \in \mathcal{M}\\
x_{ij} &\ge 0, && (i,j) \in S_T
\end{aligned}
$$

**Lemma 1.**
Every extreme-point solution of LP($T$) has at most $|\mathcal{M}| + |\mathcal{J}|$ positive variables.

**Proof.** $L(T)$: $|S_T|$ variables

# Lemma 1

$$\begin{aligned} \sum_{i:\ (i,j)\in S_T} x_{ij} &= 1, && J_j \in \mathcal{J} \\ \sum_{j:\ (i,j)\in S_T} x_{ij} p_{ij} &\le T, && M_i \in \mathcal{M} \\ x_{ij} &\ge 0, && (i,j) \in S_T \end{aligned}$$

**Lemma 1.**
Every extreme-point solution of LP($T$) has at most $|\mathcal{M}| + |\mathcal{J}|$ positive variables.

**Proof.** $L(T)$: $|S_T|$ variables
extreme-point solution: $|S_T|$ inequalities tight

# Lemma 1

$$\begin{aligned} \sum_{i\colon (i,j)\in S_T} x_{ij} &= 1, & J_j &\in \mathcal{J} \\ \sum_{j\colon (i,j)\in S_T} x_{ij} p_{ij} &\le T, & M_i &\in \mathcal{M} \\ x_{ij} &\ge 0, & (i,j) &\in S_T \end{aligned}$$

**Lemma 1.**
Every extreme-point solution of LP($T$) has at most $|\mathcal{M}| + |\mathcal{J}|$ positive variables.

**Proof.** $L(T)$: $|S_T|$ variables
extreme-point solution: $|S_T|$ inequalities tight
at most $|\mathcal{J}|$ inequalities

# Lemma 1

$$\begin{aligned} \sum_{i\colon (i,j)\in S_T} x_{ij} &= 1, & J_j &\in \mathcal{J} \\ \sum_{j\colon (i,j)\in S_T} x_{ij}p_{ij} &\le T, & M_i &\in \mathcal{M} \\ x_{ij} &\geq 0, & (i,j) &\in S_T \end{aligned}$$

**Lemma 1.**
Every extreme-point solution of LP($T$) has at most $|\mathcal{M}| + |\mathcal{J}|$ positive variables.

**Proof.** $L(T)$: $|S_T|$ variables

extreme-point solution: $|S_T|$ inequalities tight

at most $|\mathcal{J}|$ inequalities

at most $|\mathcal{M}|$ inequalities

# Lemma 1

$$\sum_{i:\,(i,j)\in S_T} x_{ij} = 1, \qquad J_j \in \mathcal{J}$$

$$\sum_{j:\,(i,j)\in S_T} x_{ij} p_{ij} \le T, \qquad M_i \in \mathcal{M}$$

$$x_{ij} \ge 0, \qquad (i,j) \in S_T$$

**Lemma 1.**
Every extreme-point solution of LP($T$) has at most $|\mathcal{M}| + |\mathcal{J}|$ positive variables.

**Proof.** $L(T)$: $|S_T|$ variables

extreme-point solution: $|S_T|$ inequalities tight

at most $|\mathcal{J}|$ inequalities

at most $|\mathcal{M}|$ inequalities

$\Rightarrow$ At least $|S_T| - |\mathcal{J}| - |\mathcal{M}|$ variables are 0.

# Lemma 1

$$\sum_{i:\,(i,j)\in S_T} x_{ij} = 1, \qquad J_j \in \mathcal{J}$$

$$\sum_{j:\,(i,j)\in S_T} x_{ij} p_{ij} \le T, \qquad M_i \in \mathcal{M}$$

$$x_{ij} \ge 0, \qquad (i,j) \in S_T$$

**Lemma 1.**
Every extreme-point solution of LP($T$) has at most $|\mathcal{M}| + |\mathcal{J}|$ positive variables.

**Proof.** $L(T)$: $|S_T|$ variables

extreme-point solution: $|S_T|$ inequalities tight

at most $|\mathcal{J}|$ inequalities

at most $|\mathcal{M}|$ inequalities

$\Rightarrow$ At least $|S_T| - |\mathcal{J}| - |\mathcal{M}|$ variables are 0.

$\Rightarrow$ At most $|\mathcal{M}| + |\mathcal{J}|$ variables are positive. □

# Lemma 2

$$\begin{aligned} \sum_{i:\,(i,j)\in S_T} x_{ij} &= 1, && J_j \in \mathcal{J} \\ \sum_{j:\,(i,j)\in S_T} x_{ij} p_{ij} &\le T, && M_i \in \mathcal{M} \\ x_{ij} &\ge 0, && (i,j) \in S_T \end{aligned}$$

**Lemma 2.**

Every extreme-point solution of LP($T$) sets at least $|\mathcal{J}|-|\mathcal{M}|$ jobs integrally.

# Lemma 2

$$\begin{aligned} \sum_{i\colon (i,j)\in S_T} x_{ij} &= 1, \quad && J_j \in \mathcal{J} \\ \sum_{j\colon (i,j)\in S_T} x_{ij} p_{ij} &\le T, \quad && M_i \in \mathcal{M} \\ x_{ij} &\ge 0, \quad && (i,j) \in S_T \end{aligned}$$

**Lemma 2.**
Every extreme-point solution of LP($T$) sets at least $|\mathcal{J}|-|\mathcal{M}|$ jobs integrally.

**Proof.** Let $x$ be an extreme-point solution of LP($T$).

# Lemma 2

$$\begin{aligned} \sum_{i:\,(i,j)\in S_T} x_{ij} &= 1, && J_j \in \mathcal{J} \\ \sum_{j:\,(i,j)\in S_T} x_{ij}p_{ij} &\le T, && M_i \in \mathcal{M} \\ x_{ij} &\ge 0, && (i,j) \in S_T \end{aligned}$$

**Lemma 2.**
Every extreme-point solution of LP($T$) sets at least $|\mathcal{J}|-|\mathcal{M}|$ jobs integrally.

**Proof.** Let $x$ be an extreme-point solution of LP($T$).
Assume $x$ has $\alpha$ integral jobs und $\beta$ fractional jobs.

# Lemma 2

$$\begin{aligned} \sum_{i\colon (i,j)\in S_T} x_{ij} &= 1, && J_j \in \mathcal{J} \\ \sum_{j\colon (i,j)\in S_T} x_{ij} p_{ij} &\le T, && M_i \in \mathcal{M} \\ x_{ij} &\ge 0, && (i,j) \in S_T \end{aligned}$$

**Lemma 2.**
Every extreme-point solution of LP($T$) sets at least $|\mathcal{J}|-|\mathcal{M}|$ jobs integrally.

**Proof.** Let $x$ be an extreme-point solution of LP($T$).
Assume $x$ has $\alpha$ integral jobs und $\beta$ fractional jobs.
$\Rightarrow \alpha + \beta = |\mathcal{J}|$

# Lemma 2

$$\begin{aligned} \sum_{i\colon (i,j)\in S_T} x_{ij} &= 1, & J_j &\in \mathcal{J} \\ \sum_{j\colon (i,j)\in S_T} x_{ij} p_{ij} &\le T, & M_i &\in \mathcal{M} \\ x_{ij} &\ge 0, & (i,j) &\in S_T \end{aligned}$$

**Lemma 2.**
Every extreme-point solution of LP($T$) sets at least $|\mathcal{J}| - |\mathcal{M}|$ jobs integrally.

**Proof.** Let $x$ be an extreme-point solution of LP($T$).
Assume $x$ has $\alpha$ integral jobs und $\beta$ fractional jobs.
$\Rightarrow \alpha + \beta = |\mathcal{J}|$
Each fractional job runs on at least two machines.

# Lemma 2

$$\begin{aligned} \sum_{i\colon (i,j)\in S_T} x_{ij} &= 1, & J_j &\in \mathcal{J} \\ \sum_{j\colon (i,j)\in S_T} x_{ij} p_{ij} &\le T, & M_i &\in \mathcal{M} \\ x_{ij} &\ge 0, & (i,j) &\in S_T \end{aligned}$$

**Lemma 2.**
Every extreme-point solution of LP($T$) sets at least $|\mathcal{J}| - |\mathcal{M}|$ jobs integrally.

**Proof.** Let $x$ be an extreme-point solution of LP($T$).
Assume $x$ has $\alpha$ integral jobs und $\beta$ fractional jobs.
$\Rightarrow \alpha + \beta = |\mathcal{J}|$
Each fractional job runs on at least two machines.
$\Rightarrow$ For each such job, at least two variables are pos.

# Lemma 2

$$\begin{aligned} \sum_{i:\,(i,j)\in S_T} x_{ij} &= 1, && J_j \in \mathcal{J} \\ \sum_{j:\,(i,j)\in S_T} x_{ij} p_{ij} &\leq T, && M_i \in \mathcal{M} \\ x_{ij} &\geq 0, && (i,j) \in S_T \end{aligned}$$

**Lemma 2.**
Every extreme-point solution of LP($T$) sets at least $|\mathcal{J}|-|\mathcal{M}|$ jobs integrally.

**Proof.** Let $x$ be an extreme-point solution of LP($T$).
Assume $x$ has $\alpha$ integral jobs und $\beta$ fractional jobs.
$\Rightarrow \alpha + \beta = |\mathcal{J}|$
Each fractional job runs on at least two machines.
$\Rightarrow$ For each such job, at least two variables are pos.
$\Rightarrow \alpha + 2\beta \leq |\mathcal{J}| + |\mathcal{M}|$ (Lemma 1)

# Lemma 2

$$\begin{aligned} \sum_{i\colon (i,j)\in S_T} x_{ij} &= 1, && J_j \in \mathcal{J} \\ \sum_{j\colon (i,j)\in S_T} x_{ij} p_{ij} &\le T, && M_i \in \mathcal{M} \\ x_{ij} &\ge 0, && (i,j) \in S_T \end{aligned}$$

**Lemma 2.**
Every extreme-point solution of LP($T$) sets at least $|\mathcal{J}|-|\mathcal{M}|$ jobs integrally.

**Proof.** Let $x$ be an extreme-point solution of LP($T$).
Assume $x$ has $\alpha$ integral jobs und $\beta$ fractional jobs.
$\Rightarrow \alpha + \beta = |\mathcal{J}|$
Each fractional job runs on at least two machines.
$\Rightarrow$ For each such job, at least two variables are pos.
$\Rightarrow \alpha + 2\beta \le |\mathcal{J}| + |\mathcal{M}|$ (Lemma 1)
$\Rightarrow \beta \le |\mathcal{M}|$

# Lemma 2

$$\begin{aligned} \sum_{i:\,(i,j)\in S_T} x_{ij} &= 1, & J_j &\in \mathcal{J} \\ \sum_{j:\,(i,j)\in S_T} x_{ij} p_{ij} &\le T, & M_i &\in \mathcal{M} \\ x_{ij} &\ge 0, & (i,j) &\in S_T \end{aligned}$$

**Lemma 2.**
Every extreme-point solution of LP($T$) sets at least $|\mathcal{J}|-|\mathcal{M}|$ jobs integrally.

**Proof.** Let $x$ be an extreme-point solution of LP($T$).
Assume $x$ has $\alpha$ integral jobs und $\beta$ fractional jobs.
$\Rightarrow \alpha + \beta = |\mathcal{J}|$
Each fractional job runs on at least two machines.
$\Rightarrow$ For each such job, at least two variables are pos.
$\Rightarrow \alpha + 2\beta \le |\mathcal{J}| + |\mathcal{M}|$ (Lemma 1)
$\Rightarrow \beta \le |\mathcal{M}|$ and $\alpha \ge |\mathcal{J}| - |\mathcal{M}|$ □

# Approximation Algorithms

## Lecture 7:
## Scheduling Jobs on Parallel Machines

### Part III:
### An Algorithm

# Extreme Point Solutions of LP($T$)

**Definition:** Bipartite graph $G = (\mathcal{M} \cup \mathcal{J}, E)$ with $(i, j) \in E \Leftrightarrow x_{ij} \neq 0$ (in extreme-point sol.).

# Extreme Point Solutions of LP($T$)

**Definition:** Bipartite graph $G = (\mathcal{M} \cup \mathcal{J}, E)$ with $(i, j) \in E \Leftrightarrow x_{ij} \neq 0$ (in extreme-point sol.).

Jobs can be assigned *integrally* or *fractionally*.

# Extreme Point Solutions of LP($T$)

**Definition:** Bipartite graph $G = (\mathcal{M} \cup \mathcal{J}, E)$ with $(i, j) \in E \Leftrightarrow x_{ij} \neq 0$ (in extreme-point sol.).

Jobs can be assigned *integrally* or *fractionally*.

$(\exists M_i \in \mathcal{M} : 0 < x_{ij} < 1)$

# Extreme Point Solutions of LP($T$)

**Definition:** Bipartite graph $G = (\mathcal{M} \cup \mathcal{J}, E)$ with $(i, j) \in E \Leftrightarrow x_{ij} \neq 0$ (in extreme-point sol.).

Jobs can be assigned *integrally* or *fractionally*.

$(\exists M_i \in \mathcal{M} : 0 < x_{ij} < 1)$

Let $F \subseteq \mathcal{J}$ be the set of fractionally assigned jobs.

# Extreme Point Solutions of LP($T$)

**Definition:** Bipartite graph $G = (\mathcal{M} \cup \mathcal{J}, E)$ with $(i, j) \in E \Leftrightarrow x_{ij} \neq 0$ (in extreme-point sol.).

Jobs can be assigned *integrally* or *fractionally*.

$(\exists M_i \in \mathcal{M} : 0 < x_{ij} < 1)$

Let $F \subseteq \mathcal{J}$ be the set of fractionally assigned jobs.
Let $H := G[\mathcal{M} \cup F]$.

# Extreme Point Solutions of LP($T$)

**Definition:** Bipartite graph $G = (\mathcal{M} \cup \mathcal{J}, E)$ with $(i, j) \in E \Leftrightarrow x_{ij} \neq 0$ (in extreme-point sol.).

Jobs can be assigned *integrally* or *fractionally*.
$(\exists M_i \in \mathcal{M} : 0 < x_{ij} < 1)$

Let $F \subseteq \mathcal{J}$ be the set of fractionally assigned jobs.
Let $H := G[\mathcal{M} \cup F]$.

**Observe:** $(i, j)$ is an edge in $H \Leftrightarrow 0 < x_{ij} < 1$

# Extreme Point Solutions of LP($T$)

**Definition:** Bipartite graph $G = (\mathcal{M} \cup \mathcal{J}, E)$ with $(i, j) \in E \Leftrightarrow x_{ij} \neq 0$ (in extreme-point sol.).

Jobs can be assigned *integrally* or *fractionally*.

$(\exists M_i \in \mathcal{M} \colon 0 < x_{ij} < 1)$

Let $F \subseteq \mathcal{J}$ be the set of fractionally assigned jobs.
Let $H := G[\mathcal{M} \cup F]$.

**Observe:** $(i, j)$ is an edge in $H \Leftrightarrow 0 < x_{ij} < 1$

A matching in $H$ is called $F$*-perfect* if it matches every vertex in $F$.

# Extreme Point Solutions of LP($T$)

**Definition:** Bipartite graph $G = (\mathcal{M} \cup \mathcal{J}, E)$ with $(i,j) \in E \Leftrightarrow x_{ij} \neq 0$ (in extreme-point sol.).

Jobs can be assigned *integrally* or *fractionally*.
$(\exists M_i \in \mathcal{M} \colon 0 < x_{ij} < 1)$

Let $F \subseteq \mathcal{J}$ be the set of fractionally assigned jobs.
Let $H := G[\mathcal{M} \cup F]$.

**Observe:** $(i,j)$ is an edge in $H \Leftrightarrow 0 < x_{ij} < 1$

A matching in $H$ is called $F$*-perfect* if it matches every vertex in $F$.

**Main step:** Show that $H$ always has an $F$-perfect matching.

# Extreme Point Solutions of LP($T$)

**Definition:** Bipartite graph $G = (\mathcal{M} \cup \mathcal{J}, E)$ with $(i, j) \in E \Leftrightarrow x_{ij} \neq 0$ (in extreme-point sol.).

Jobs can be assigned *integrally* or *fractionally*.
$(\exists M_i \in \mathcal{M} \colon 0 < x_{ij} < 1)$

Let $F \subseteq \mathcal{J}$ be the set of fractionally assigned jobs.
Let $H := G[\mathcal{M} \cup F]$.

**Observe:** $(i, j)$ is an edge in $H \Leftrightarrow 0 < x_{ij} < 1$

A matching in $H$ is called $F$-*perfect* if it matches every vertex in $F$.

**Main step:** Show that $H$ always has an $F$-perfect matching.

And why is this useful . . . ?

# Algorithm

Assign job $J_j$ to machine $M_i$ that minimizes $p_{ij}$.

# Algorithm

Assign job $J_j$ to machine $M_i$ that minimizes $p_{ij}$.
Let $\tau$ be the makespan of this schedule.

# Algorithm

Assign job $J_j$ to machine $M_i$ that minimizes $p_{ij}$.
Let $\tau$ be the makespan of this schedule.

Do a binary search in the interval $[\frac{\tau}{|\mathcal{M}|}, \tau]$ to find the smallest value $T^*$ of $T \in \mathbb{Z}^+$ s.t. LP($T$) has a feasible solution.

# Algorithm

Assign job $J_j$ to machine $M_i$ that minimizes $p_{ij}$.
Let $\tau$ be the makespan of this schedule.

Do a binary search in the interval $[\frac{\tau}{|\mathcal{M}|}, \tau]$ to find the smallest value $T^*$ of $T \in \mathbb{Z}^+$ s.t. LP($T$) has a feasible solution.

Find an extreme-point solution $x$ for LP($T^*$).

# Algorithm

Assign job $J_j$ to machine $M_i$ that minimizes $p_{ij}$.
Let $\tau$ be the makespan of this schedule.

Do a binary search in the interval $[\frac{\tau}{|\mathcal{M}|}, \tau]$ to find the smallest value $T^*$ of $T \in \mathbb{Z}^+$ s.t. LP($T$) has a feasible solution.

Find an extreme-point solution $x$ for LP($T^*$).

Assign all integrally set jobs to machines as in $x$.

# Algorithm

Assign job $J_j$ to machine $M_i$ that minimizes $p_{ij}$.
Let $\tau$ be the makespan of this schedule.

Do a binary search in the interval $[\frac{\tau}{|\mathcal{M}|}, \tau]$ to find the smallest value $T^*$ of $T \in \mathbb{Z}^+$ s.t. LP($T$) has a feasible solution.

Find an extreme-point solution $x$ for LP($T^*$).

Assign all integrally set jobs to machines as in $x$.

Construct the graph $H$ and find an $F$-perfect matching $P$ in it (see Lemma 4 later, $F$ is set of fractionally assg. jobs)

# Algorithm

Assign job $J_j$ to machine $M_i$ that minimizes $p_{ij}$.
Let $\tau$ be the makespan of this schedule.

Do a binary search in the interval $[\frac{\tau}{|\mathcal{M}|}, \tau]$ to find the smallest value $T^*$ of $T \in \mathbb{Z}^+$ s.t. LP($T$) has a feasible solution.

Find an extreme-point solution $x$ for LP($T^*$).

Assign all integrally set jobs to machines as in $x$.

Construct the graph $H$ and find an $F$-perfect matching $P$ in it (see Lemma 4 later, $F$ is set of fractionally assg. jobs)

Assign the fractional jobs to machines using $P$.

# Algorithm

Assign job $J_j$ to machine $M_i$ that minimizes $p_{ij}$.
Let $\tau$ be the makespan of this schedule.

Do a binary search in the interval $[\frac{\tau}{|\mathcal{M}|}, \tau]$ to find the smallest value $T^*$ of $T \in \mathbb{Z}^+$ s.t. LP($T$) has a feasible solution.

Find an extreme-point solution $x$ for LP($T^*$).

Assign all integrally set jobs to machines as in $x$.

Construct the graph $H$ and find an $F$-perfect matching $P$ in it (see Lemma 4 later, $F$ is set of fractionally assg. jobs)

Assign the fractional jobs to machines using $P$.

**Theorem.** This is a factor-2 approximation algorithm (assuming that we have an $F$-perfect matching).

# Approximation Factor

$$\begin{aligned} \sum_{i\colon (i,j)\in S_T} x_{ij} &= 1, & J_j &\in \mathcal{J} \\ \sum_{j\colon (i,j)\in S_T} x_{ij} p_{ij} &\le T, & M_i &\in \mathcal{M} \\ x_{ij} &\ge 0, & (i,j) &\in S_T \end{aligned}$$

**Theorem.** This is a factor-2 approximation algorithm (assuming that we have an $F$-perfect matching).

**Proof.** $T^* \le \mathrm{OPT}$.

# Approximation Factor

$$\begin{aligned} \sum_{i\colon (i,j)\in S_T} x_{ij} &= 1, & J_j &\in \mathcal{J} \\ \sum_{j\colon (i,j)\in S_T} x_{ij} p_{ij} &\le T, & M_i &\in \mathcal{M} \\ x_{ij} &\ge 0, & (i,j) &\in S_T \end{aligned}$$

**Theorem.** This is a factor-2 approximation algorithm (assuming that we have an $F$-perfect matching).

**Proof.** $T^* \le \mathrm{OPT}$.

Let $x$ be an extreme-point solution for $LP(T^*)$

# Approximation Factor

$$\begin{aligned} \sum_{i\colon (i,j)\in S_T} x_{ij} &= 1, & J_j &\in \mathcal{J} \\ \sum_{j\colon (i,j)\in S_T} x_{ij} p_{ij} &\le T, & M_i &\in \mathcal{M} \\ x_{ij} &\ge 0, & (i,j) &\in S_T \end{aligned}$$

**Theorem.** This is a factor-2 approximation algorithm (assuming that we have an $F$-perfect matching).

**Proof.** $T^* \le \mathrm{OPT}$.

Let $x$ be an extreme-point solution for $LP(T^*)$

Fractional solution: makespan $\le T^*$.

# Approximation Factor

$$\sum_{i\colon (i,j)\in S_T} x_{ij} = 1, \quad J_j \in \mathcal{J}$$

$$\sum_{j\colon (i,j)\in S_T} x_{ij} p_{ij} \le T, \quad M_i \in \mathcal{M}$$

$$x_{ij} \ge 0, \quad (i,j) \in S_T$$

**Theorem.** This is a factor-2 approximation algorithm (assuming that we have an $F$-perfect matching).

**Proof.** $T^* \le \mathrm{OPT}$.

Let $x$ be an extreme-point solution for $LP(T^*)$

Fractional solution: makespan $\le T^*$.

$\Rightarrow$ Restriction to integral jobs has makespan $\le T^*$.

# Approximation Factor

$$\begin{array}{rll} \displaystyle\sum_{i\colon (i,j)\in S_T} x_{ij} = 1, & J_j \in \mathcal{J} \\ \displaystyle\sum_{j\colon (i,j)\in S_T} x_{ij} p_{ij} \leq T, & M_i \in \mathcal{M} \\ x_{ij} \geq 0, & (i,j) \in S_T \end{array}$$

**Theorem.** This is a factor-2 approximation algorithm (assuming that we have an $F$-perfect matching).

**Proof.** $T^* \leq \text{OPT}$.

Let $x$ be an extreme-point solution for $LP(T^*)$

Fractional solution: makespan $\leq T^*$.

$\Rightarrow$ Restriction to integral jobs has makespan $\leq T^*$.

For each edge $(i,j) \in S_{T^*}$, it holds that $p_{ij} \leq T^*$.

# Approximation Factor

$$\begin{aligned} \sum_{i\colon (i,j)\in S_T} x_{ij} &= 1, && J_j \in \mathcal{J} \\ \sum_{j\colon (i,j)\in S_T} x_{ij} p_{ij} &\le T, && M_i \in \mathcal{M} \\ x_{ij} &\ge 0, && (i,j) \in S_T \end{aligned}$$

**Theorem.** This is a factor-2 approximation algorithm (assuming that we have an $F$-perfect matching).

**Proof.** $T^* \le \mathrm{OPT}$.
Let $x$ be an extreme-point solution for $LP(T^*)$
Fractional solution: makespan $\le T^*$.
$\Rightarrow$ Restriction to integral jobs has makespan $\le T^*$.
For each edge $(i,j) \in S_{T^*}$, it holds that $p_{ij} \le T^*$.
Matching: at most one extra job per maschine.

# Approximation Factor

$$\sum_{i:\,(i,j)\in S_T} x_{ij} = 1, \quad J_j \in \mathcal{J}$$

$$\sum_{j:\,(i,j)\in S_T} x_{ij} p_{ij} \le T, \quad M_i \in \mathcal{M}$$

$$x_{ij} \ge 0, \quad (i,j) \in S_T$$

**Theorem.** This is a factor-2 approximation algorithm (assuming that we have an $F$-perfect matching).

**Proof.** $T^* \le \mathrm{OPT}$.

Let $x$ be an extreme-point solution for $LP(T^*)$

Fractional solution: makespan $\le T^*$.

$\Rightarrow$ Restriction to integral jobs has makespan $\le T^*$.

For each edge $(i,j) \in S_{T^*}$, it holds that $p_{ij} \le T^*$.

Matching: at most one extra job per maschine.

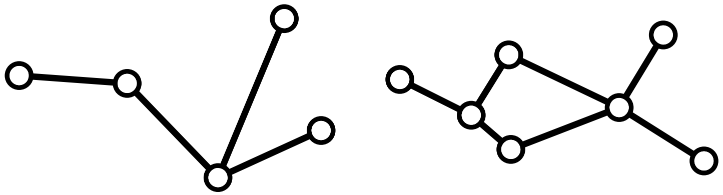
$\Rightarrow$ total makespan $\le 2T^* \le 2\mathrm{OPT}$ □

# Approximation Algorithms

## Lecture 7:
## Scheduling Jobs on Parallel Machines

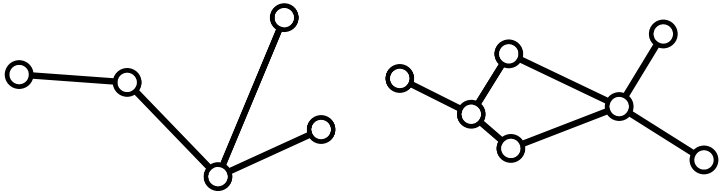
### Part IV:
### Pseudo-Trees and -Forests

# Pseudo-Trees and -Forests



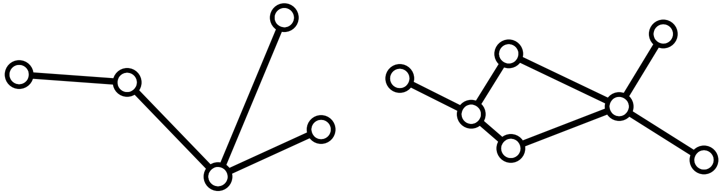
**Pseudo-tree**: a connected graph with at most as many edges as vertices.

# Pseudo-Trees and -Forests



**Pseudo-tree**: a connected graph with at most as many edges as vertices.
(A pseudo-tree is either a tree or a tree plus a single edge.)

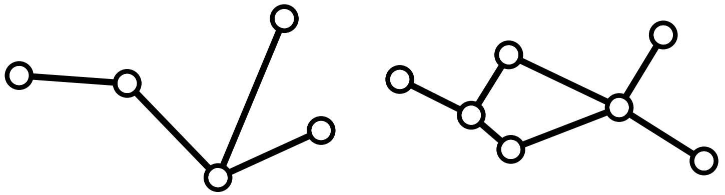
# Pseudo-Trees and -Forests



**Pseudo-tree**: a connected graph with at most as many edges as vertices.
(A pseudo-tree is either a tree or a tree plus a single edge.)

**Pseudo-forest**:

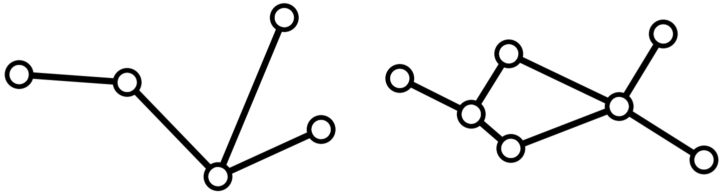
# Pseudo-Trees and -Forests



**Pseudo-tree**: a connected graph with at most as many edges as vertices.
(A pseudo-tree is either a tree or a tree plus a single edge.)

**Pseudo-forest**: a collection of disjoint pseudo-trees.

# Pseudo-Trees and -Forests

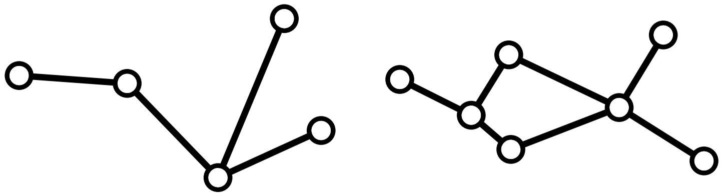


**Pseudo-tree**: a connected graph with at most as many edges as vertices.
(A pseudo-tree is either a tree or a tree plus a single edge.)

**Pseudo-forest**: a collection of disjoint pseudo-trees.

**Lemma 3.**
The bipartite graph $G = (\mathcal{M} \cup \mathcal{J}, E)$ is a pseudo-forest.

# Pseudo-Trees and -Forests



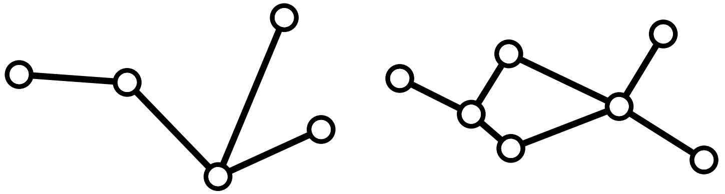
**Pseudo-tree**: a connected graph with at most as many edges as vertices. (A pseudo-tree is either a tree or a tree plus a single edge.)

**Pseudo-forest**: a collection of disjoint pseudo-trees.

**Lemma 3.**
The bipartite graph $G = (\mathcal{M} \cup \mathcal{J}, E)$ is a pseudo-forest.

Extreme-point solutions have $\leq |\mathcal{M}| + |\mathcal{J}|$ positive variables (Lemma 1).

# Pseudo-Trees and -Forests



**Pseudo-tree**: a connected graph with at most as many edges as vertices. (A pseudo-tree is either a tree or a tree plus a single edge.)

**Pseudo-forest**: a collection of disjoint pseudo-trees.

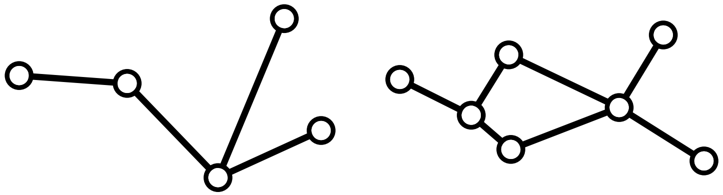
**Lemma 3.**
The bipartite graph $G = (\mathcal{M} \cup \mathcal{J}, E)$ is a pseudo-forest.

Extreme-point solutions have $\leq |\mathcal{M}| + |\mathcal{J}|$ positive variables (Lemma 1).

Each conn. component $C$ of $G$ corresponds to an extreme-point solution.

(Suppose not. Then the solution that corresponds to $C$ is the convex combination of other solutions. But this contradicts the definition of $G$.)

# Pseudo-Trees and -Forests



**Pseudo-tree**: a connected graph with at most as many edges as vertices. (A pseudo-tree is either a tree or a tree plus a single edge.)

**Pseudo-forest**: a collection of disjoint pseudo-trees.

**Lemma 3.**
The bipartite graph $G = (\mathcal{M} \cup \mathcal{J}, E)$ is a pseudo-forest.

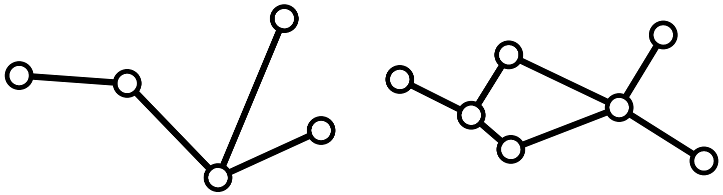
Extreme-point solutions have $\leq |\mathcal{M}| + |\mathcal{J}|$ positive variables (Lemma 1).

Each conn. component $C$ of $G$ corresponds to an extreme-point solution.

(Suppose not. Then the solution that corresponds to $C$ is the convex combination of other solutions. But this contradicts the definition of $G$.)

$\Rightarrow$ $C$ has at most as many edges (pos. var.) as vertices (jobs+machines).

# Pseudo-Trees and -Forests



**Pseudo-tree**: a connected graph with at most as many edges as vertices.
(A pseudo-tree is either a tree or a tree plus a single edge.)

**Pseudo-forest**: a collection of disjoint pseudo-trees.

**Lemma 3.**
The bipartite graph $G = (\mathcal{M} \cup \mathcal{J}, E)$ is a pseudo-forest.

Extreme-point solutions have $\le |\mathcal{M}| + |\mathcal{J}|$ positive variables (Lemma 1).

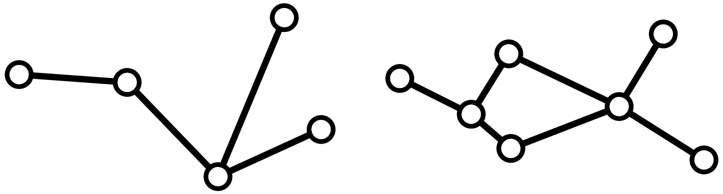
Each conn. component $C$ of $G$ corresponds to an extreme-point solution.

(Suppose not. Then the solution that corresponds to $C$ is the convex combination of other solutions. But this contradicts the definition of $G$.)

$\Rightarrow$ $C$ has at most as many edges (pos. var.) as vertices (jobs+machines).

**Lemma 4.** The graph $H$ has an $F$-perfect matching.

# Pseudo-Trees and -Forests



**Pseudo-tree**: a connected graph with at most as many edges as vertices.
(A pseudo-tree is either a tree or a tree plus a single edge.)

**Pseudo-forest**: a collection of disjoint pseudo-trees.

**Lemma 3.**
The bipartite graph $G = (\mathcal{M} \cup \mathcal{J}, E)$ is a pseudo-forest.

Extreme-point solutions have $\leq |\mathcal{M}| + |\mathcal{J}|$ positive variables (Lemma 1).

Each conn. component $C$ of $G$ corresponds to an extreme-point solution.

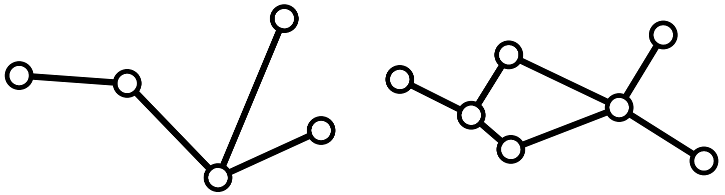
(Suppose not. Then the solution that corresponds to $C$ is the convex combination of other solutions. But this contradicts the definition of $G$.)

$\Rightarrow$ $C$ has at most as many edges (pos. var.) as vertices (jobs+machines).

**Lemma 4.** The graph $H$ has an $F$-perfect matching.

In $G$, every vertex in $\mathcal{J} \setminus F$ is a leaf. $\Rightarrow$

# Pseudo-Trees and -Forests



**Pseudo-tree**: a connected graph with at most as many edges as vertices. (A pseudo-tree is either a tree or a tree plus a single edge.)

**Pseudo-forest**: a collection of disjoint pseudo-trees.

**Lemma 3.**
The bipartite graph $G = (\mathcal{M} \cup \mathcal{J}, E)$ is a pseudo-forest.

Extreme-point solutions have $\leq |\mathcal{M}| + |\mathcal{J}|$ positive variables (Lemma 1).

Each conn. component $C$ of $G$ corresponds to an extreme-point solution.

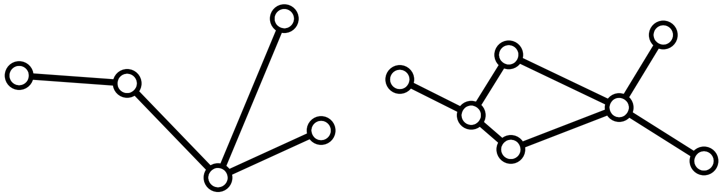
(Suppose not. Then the solution that corresponds to $C$ is the convex combination of other solutions. But this contradicts the definition of $G$.)

$\Rightarrow$ $C$ has at most as many edges (pos. var.) as vertices (jobs+machines).

**Lemma 4.** The graph $H$ has an $F$-perfect matching.

In $G$, every vertex in $\mathcal{J} \setminus F$ is a leaf. $\overset{\text{remove leaves}}{\Rightarrow}$

# Pseudo-Trees and -Forests



**Pseudo-tree**: a connected graph with at most as many edges as vertices. (A pseudo-tree is either a tree or a tree plus a single edge.)

**Pseudo-forest**: a collection of disjoint pseudo-trees.

**Lemma 3.**
The bipartite graph $G = (\mathcal{M} \cup \mathcal{J}, E)$ is a pseudo-forest.

Extreme-point solutions have $\le |\mathcal{M}| + |\mathcal{J}|$ positive variables (Lemma 1).

Each conn. component $C$ of $G$ corresponds to an extreme-point solution.

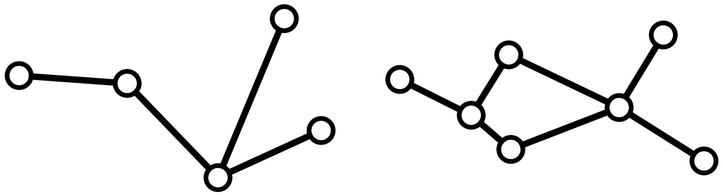
(Suppose not. Then the solution that corresponds to $C$ is the convex combination of other solutions. But this contradicts the definition of $G$.)

$\Rightarrow$ $C$ has at most as many edges (pos. var.) as vertices (jobs+machines).

**Lemma 4.** The graph $H$ has an $F$-perfect matching.

In $G$, every vertex in $\mathcal{J} \setminus F$ is a leaf. $\overset{\text{remove leaves}}{\Rightarrow}$ $H$ is a pseudo-forest, too.

# Pseudo-Trees and -Forests



**Pseudo-tree**: a connected graph with at most as many edges as vertices.
(A pseudo-tree is either a tree or a tree plus a single edge.)

**Pseudo-forest**: a collection of disjoint pseudo-trees.

**Lemma 3.**
The bipartite graph $G = (\mathcal{M} \cup \mathcal{J}, E)$ is a pseudo-forest.

Extreme-point solutions have $\leq |\mathcal{M}| + |\mathcal{J}|$ positive variables (Lemma 1).

Each conn. component $C$ of $G$ corresponds to an extreme-point solution.

(Suppose not. Then the solution that corresponds to $C$ is the convex combination of other solutions. But this contradicts the definition of $G$.)

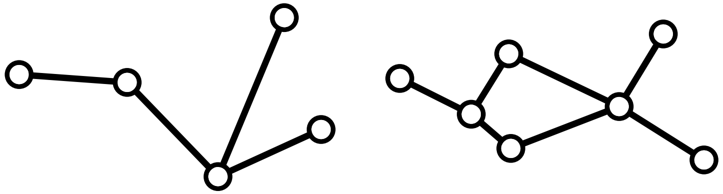
$\Rightarrow$ $C$ has at most as many edges (pos. var.) as vertices (jobs+machines).

**Lemma 4.** The graph $H$ has an $F$-perfect matching.

In $G$, every vertex in $\mathcal{J} \setminus F$ is a leaf. $\overset{\text{remove leaves}}{\Rightarrow}$ $H$ is a pseudo-forest, too.

Vertices in $F$ have minimum degree 2.

# Pseudo-Trees and -Forests



**Pseudo-tree**: a connected graph with at most as many edges as vertices. (A pseudo-tree is either a tree or a tree plus a single edge.)

**Pseudo-forest**: a collection of disjoint pseudo-trees.

**Lemma 3.**
The bipartite graph $G = (\mathcal{M} \cup \mathcal{J}, E)$ is a pseudo-forest.

Extreme-point solutions have $\leq |\mathcal{M}| + |\mathcal{J}|$ positive variables (Lemma 1).

Each conn. component $C$ of $G$ corresponds to an extreme-point solution.

(Suppose not. Then the solution that corresponds to $C$ is the convex combination of other solutions. But this contradicts the definition of $G$.)

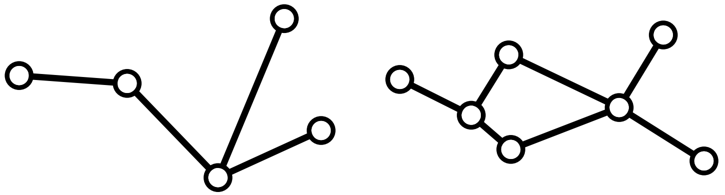
$\Rightarrow$ $C$ has at most as many edges (pos. var.) as vertices (jobs+machines).

**Lemma 4.** The graph $H$ has an $F$-perfect matching.

In $G$, every vertex in $\mathcal{J} \setminus F$ is a leaf. $\overset{\text{remove leaves}}{\Rightarrow}$ $H$ is a pseudo-forest, too.

Vertices in $F$ have minimum degree 2. $\Rightarrow$ The leaves in $H$ are machines.

# Pseudo-Trees and -Forests



**Pseudo-tree**: a connected graph with at most as many edges as vertices.
(A pseudo-tree is either a tree or a tree plus a single edge.)

**Pseudo-forest**: a collection of disjoint pseudo-trees.

**Lemma 3.**
The bipartite graph $G = (\mathcal{M} \cup \mathcal{J}, E)$ is a pseudo-forest.

Extreme-point solutions have $\leq |\mathcal{M}| + |\mathcal{J}|$ positive variables (Lemma 1).

Each conn. component $C$ of $G$ corresponds to an extreme-point solution.

(Suppose not. Then the solution that corresponds to $C$ is the convex combination of other solutions. But this contradicts the definition of $G$.)

$\Rightarrow$ $C$ has at most as many edges (pos. var.) as vertices (jobs+machines).

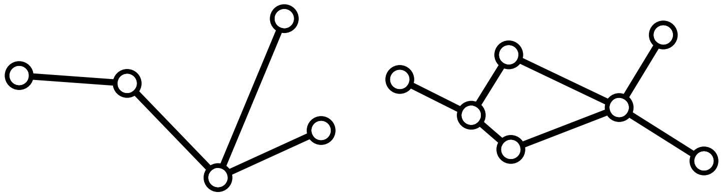
**Lemma 4.** The graph $H$ has an $F$-perfect matching.

In $G$, every vertex in $\mathcal{J} \setminus F$ is a leaf. $\overset{\text{remove leaves}}{\Rightarrow}$ $H$ is a pseudo-forest, too.

Vertices in $F$ have minimum degree 2. $\Rightarrow$ The leaves in $H$ are machines.

After iteratively matching all leaves,

# Pseudo-Trees and -Forests



**Pseudo-tree**: a connected graph with at most as many edges as vertices.
(A pseudo-tree is either a tree or a tree plus a single edge.)

**Pseudo-forest**: a collection of disjoint pseudo-trees.

**Lemma 3.**
The bipartite graph $G = (\mathcal{M} \cup \mathcal{J}, E)$ is a pseudo-forest.

Extreme-point solutions have $\le |\mathcal{M}| + |\mathcal{J}|$ positive variables (Lemma 1).

Each conn. component $C$ of $G$ corresponds to an extreme-point solution.

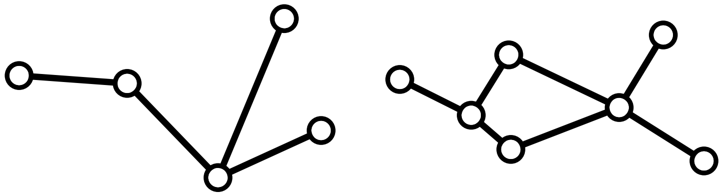
(Suppose not. Then the solution that corresponds to $C$ is the convex combination of other solutions. But this contradicts the definition of $G$.)

$\Rightarrow$ $C$ has at most as many edges (pos. var.) as vertices (jobs+machines).

**Lemma 4.** The graph $H$ has an $F$-perfect matching.

In $G$, every vertex in $\mathcal{J} \setminus F$ is a leaf. $\overset{\text{remove leaves}}{\Rightarrow}$ $H$ is a pseudo-forest, too.
Vertices in $F$ have minimum degree 2. $\Rightarrow$ The leaves in $H$ are machines.
After iteratively matching all leaves, only *even* cycles remain.

# Pseudo-Trees and -Forests



**Pseudo-tree**: a connected graph with at most as many edges as vertices.
(A pseudo-tree is either a tree or a tree plus a single edge.)

**Pseudo-forest**: a collection of disjoint pseudo-trees.

**Lemma 3.**
The bipartite graph $G = (\mathcal{M} \cup \mathcal{J}, E)$ is a pseudo-forest.

Extreme-point solutions have $\leq |\mathcal{M}| + |\mathcal{J}|$ positive variables (Lemma 1).

Each conn. component $C$ of $G$ corresponds to an extreme-point solution.

(Suppose not. Then the solution that corresponds to $C$ is the convex combination of other solutions. But this contradicts the definition of $G$.)

$\Rightarrow$ $C$ has at most as many edges (pos. var.) as vertices (jobs+machines).

**Lemma 4.** The graph $H$ has an $F$-perfect matching.

In $G$, every vertex in $\mathcal{J} \setminus F$ is a leaf. $\overset{\text{remove leaves}}{\Rightarrow}$ $H$ is a pseudo-forest, too.

Vertices in $F$ have minimum degree 2. $\Rightarrow$ The leaves in $H$ are machines.

After iteratively matching all leaves, only *even* cycles remain. ($H$ is bipartite :-)

# Scheduling on Parallel Machines

**Theorem.** There is an LP-based 2-approximation algorithm for the problem of scheduling jobs on unrelated parallel machines.

# Scheduling on Parallel Machines

**Theorem.** There is an LP-based 2-approximation algorithm for the problem of scheduling jobs on unrelated parallel machines.

Tight?

# Scheduling on Parallel Machines

**Theorem.** There is an LP-based 2-approximation algorithm for the problem of scheduling jobs on unrelated parallel machines.

Tight? Yes!

**Instance** $I_m$:

# Scheduling on Parallel Machines

**Theorem.** There is an LP-based 2-approximation algorithm for the problem of scheduling jobs on unrelated parallel machines.

Tight? Yes!

**Instance** $I_m$:

$m$ machines and $m^2 - m + 1$ jobs

# Scheduling on Parallel Machines

**Theorem.** There is an LP-based 2-approximation algorithm for the problem of scheduling jobs on unrelated parallel machines.

Tight? Yes!

**Instance** $I_m$:

$m$ machines and $m^2 - m + 1$ jobs

Job $J_1$ has processing time $m$ on every machine,

# Scheduling on Parallel Machines

**Theorem.** There is an LP-based 2-approximation algorithm for the problem of scheduling jobs on unrelated parallel machines.

Tight? Yes!

**Instance** $I_m$:

$m$ machines and $m^2 - m + 1$ jobs

Job $J_1$ has processing time $m$ on every machine,

all other jobs have processing time 1 on every machine.

# Scheduling on Parallel Machines

**Theorem.** There is an LP-based 2-approximation algorithm for the problem of scheduling jobs on unrelated parallel machines.

Tight? Yes!

**Instance** $I_m$:

$m$ machines and $m^2 - m + 1$ jobs

Job $J_1$ has processing time $m$ on every machine,

all other jobs have processing time 1 on every machine.

**Optimum:**

# Scheduling on Parallel Machines

**Theorem.** There is an LP-based 2-approximation algorithm for the problem of scheduling jobs on unrelated parallel machines.

Tight? Yes!

**Instance** $I_m$:

$m$ machines and $m^2 - m + 1$ jobs

Job $J_1$ has processing time $m$ on every machine,

all other jobs have processing time 1 on every machine.

**Optimum:** one machine gets $J_1$, and all others spread evenly.

# Scheduling on Parallel Machines

**Theorem.** There is an LP-based 2-approximation algorithm for the problem of scheduling jobs on unrelated parallel machines.

Tight? Yes!

**Instance** $I_m$:

$m$ machines and $m^2 - m + 1$ jobs

Job $J_1$ has processing time $m$ on every machine,

all other jobs have processing time 1 on every machine.

**Optimum:** one machine gets $J_1$, and all others spread evenly.

$\Rightarrow$ makespan $= m$.

# Scheduling on Parallel Machines

**Theorem.** There is an LP-based 2-approximation algorithm for the problem of scheduling jobs on unrelated parallel machines.

Tight? Yes!

**Instance $I_m$:**

$m$ machines and $m^2 - m + 1$ jobs

Job $J_1$ has processing time $m$ on every machine,

all other jobs have processing time 1 on every machine.

**Optimum:** one machine gets $J_1$, and all others spread evenly.

$\Rightarrow$ makespan $= m$.

**Algorithm:**

# Scheduling on Parallel Machines

**Theorem.** There is an LP-based 2-approximation algorithm for the problem of scheduling jobs on unrelated parallel machines.

Tight? Yes!

**Instance** $I_m$:

$m$ machines and $m^2 - m + 1$ jobs

Job $J_1$ has processing time $m$ on every machine,

all other jobs have processing time 1 on every machine.

**Optimum:** one machine gets $J_1$, and all others spread evenly.

$\Rightarrow$ makespan $= m$.

**Algorithm:**

LP($T$) has no feasible solution for any $T < m$.

# Scheduling on Parallel Machines

**Theorem.** There is an LP-based 2-approximation algorithm for the problem of scheduling jobs on unrelated parallel machines.

Tight? Yes!

**Instance $I_m$:**

$m$ machines and $m^2 - m + 1$ jobs

Job $J_1$ has processing time $m$ on every machine,

all other jobs have processing time 1 on every machine.

**Optimum:** one machine gets $J_1$, and all others spread evenly.

$\Rightarrow$ makespan $= m$.

**Algorithm:**

LP($T$) has no feasible solution for any $T < m$.

Extreme-point solution:

Assign $1/m$ of $J_1$ and $m-1$ other jobs to each machine.

# Scheduling on Parallel Machines

**Theorem.** There is an LP-based 2-approximation algorithm for the problem of scheduling jobs on unrelated parallel machines.

Tight? Yes!

**Instance $I_m$:**

$m$ machines and $m^2 - m + 1$ jobs

Job $J_1$ has processing time $m$ on every machine,

all other jobs have processing time 1 on every machine.

**Optimum:** one machine gets $J_1$, and all others spread evenly.

$\Rightarrow$ makespan $= m$.

**Algorithm:**

LP($T$) has no feasible solution for any $T < m$.

Extreme-point solution:

Assign $1/m$ of $J_1$ and $m - 1$ other jobs to each machine.

$\Rightarrow$ makespan $2m - 1$.

# Scheduling on Parallel Machines

**Theorem.** There is an LP-based 2-approximation algorithm for the problem of scheduling jobs on unrelated parallel machines.

**Can we do better?**

# Scheduling on Parallel Machines

**Theorem.** There is an LP-based 2-approximation algorithm for the problem of scheduling jobs on unrelated parallel machines.

**Can we do better?**

No better approximation algorithm is known.

# Scheduling on Parallel Machines

**Theorem.** There is an LP-based 2-approximation algorithm for the problem of scheduling jobs on unrelated parallel machines.

**Can we do better?**

No better approximation algorithm is known.

The problem cannot be approximated within factor $< 3/2$ (unless P$=$NP). [Lenstra, Shmoys & Tardos '90]

# Scheduling on Parallel Machines

**Theorem.** There is an LP-based $2$-approximation algorithm for the problem of scheduling jobs on unrelated parallel machines.

**Can we do better?**

No better approximation algorithm is known.

The problem cannot be approximated within factor $< 3/2$ (unless P$=$NP). [Lenstra, Shmoys & Tardos '90]

For a constant number of machines, for every $\varepsilon > 0$ there is a factor-$(1 + \varepsilon)$ approximation algorithm. [Horowitz & Sahni '76]

# Scheduling on Parallel Machines

**Theorem.** There is an LP-based $2$-approximation algorithm for the problem of scheduling jobs on unrelated parallel machines.

**Can we do better?**

No better approximation algorithm is known.

The problem cannot be approximated within factor $< 3/2$ (unless P$=$NP). [Lenstra, Shmoys & Tardos '90]

For a constant number of machines, for every $\varepsilon > 0$ there is a factor-$(1+\varepsilon)$ approximation algorithm. [Horowitz & Sahni '76]

For uniform machines, for every $\varepsilon > 0$ there is a factor-$(1+\varepsilon)$ approximation algorithm. [Hochbaum & Shmoys '87]

# Scheduling on Parallel Machines

**Theorem.** There is an LP-based $2$-approximation algorithm for the problem of scheduling jobs on unrelated parallel machines.

**Can we do better?**

No better approximation algorithm is known.

The problem cannot be approximated within factor $< 3/2$ (unless P$=$NP). [Lenstra, Shmoys & Tardos '90]

For a constant number of machines, for every $\varepsilon > 0$ there is a factor-$(1+\varepsilon)$ approximation algorithm. [Horowitz & Sahni '76]

For uniform machines, for every $\varepsilon > 0$ there is a factor-$(1+\varepsilon)$ approximation algorithm. [Hochbaum & Shmoys '87]

(Machines may have different speeds, but process jobs uniformly.)