Julius-Maximilians-
UNIVERSITÄT
WÜRZBURG

Chair for
INFORMATICS I
Efficient Algorithms and
Knowledge-Based Systems

Institute for Informatics

Approximationsalgorithmen

Ablaufplanung auf parallelen Maschinen durch parametrisiertes Stutzen

7. Vorlesung

Philipp Kindermann

Wintersemester 2019/20

Ablaufplanung auf parallelen Maschinen

Gegeben: Eine Menge $\mathcal{A}$ von **Aufträgen**,

Ablaufplanung auf parallelen Maschinen

Gegeben: Eine Menge $\mathcal{A}$ von **Aufträgen**,
eine Menge $\mathcal{M}$ von **Machinen**, und

Ablaufplanung auf parallelen Maschinen

Gegeben: Eine Menge $\mathcal{A}$ von **Aufträgen**,
eine Menge $\mathcal{M}$ von **Machinen**, und
für jedes $M_i \in \mathcal{M}$ und $A_j \in \mathcal{A}$ die
Bearbeitungszeit $p_{ij} \in \mathbb{N}^+$ von A_j auf M_i

Ablaufplanung auf parallelen Maschinen

Gegeben: Eine Menge $\mathcal{A}$ von **Aufträgen**,
eine Menge $\mathcal{M}$ von **Machinen**, und
für jedes $M_i \in \mathcal{M}$ und $A_j \in \mathcal{A}$ die
Bearbeitungszeit $p_{ij} \in \mathbb{N}^+$ von A_j auf M_i

$$\mathcal{A} = \{A_1, A_2, \ldots, A_8\}$$

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

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

Ablaufplanung auf parallelen Maschinen

Gegeben: Eine Menge $\mathcal{A}$ von **Aufträgen**,
eine Menge $\mathcal{M}$ von **Machinen**, und
für jedes $M_i \in \mathcal{M}$ und $A_j \in \mathcal{A}$ die
Bearbeitungszeit $p_{ij} \in \mathbb{N}^+$ von A_j auf M_i

Gesucht: Ein **Ablaufplan** $\sigma\colon \mathcal{A} \to \mathcal{M}$ der Aufträge auf den Maschinen, der die **Bearbeitungsdauer**, also die maximale Zeit, die eine Maschine in Gebrauch ist, minimiert.

$$\mathcal{A} = \{A_1, A_2, \ldots, A_8\}$$

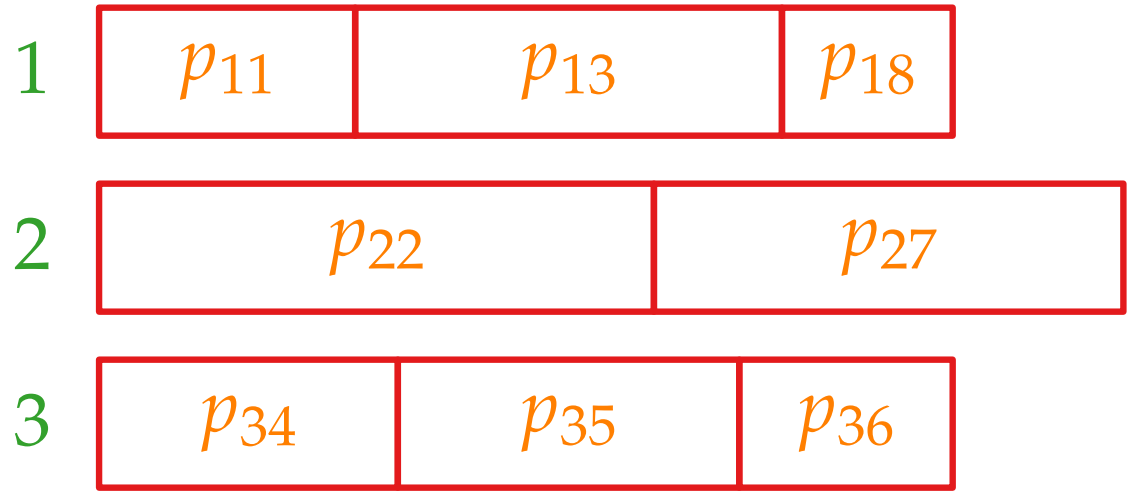
$$\mathcal{M} = \{M_1, M_2, M_3\}$$

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

Ablaufplanung auf parallelen Maschinen

Gegeben: Eine Menge $\mathcal{A}$ von **Aufträgen**,
eine Menge $\mathcal{M}$ von **Machinen**, und
für jedes $M_i \in \mathcal{M}$ und $A_j \in \mathcal{A}$ die
Bearbeitungszeit $p_{ij} \in \mathbb{N}^+$ von A_j auf M_i

Gesucht: Ein **Ablaufplan** $\sigma\colon \mathcal{A} \to \mathcal{M}$ der Aufträge auf den Maschinen, der die **Bearbeitungsdauer**, also die maximale Zeit, die eine Maschine in Gebrauch ist, minimiert.



$$\mathcal{A} = \{A_1, A_2, \ldots, A_8\}$$

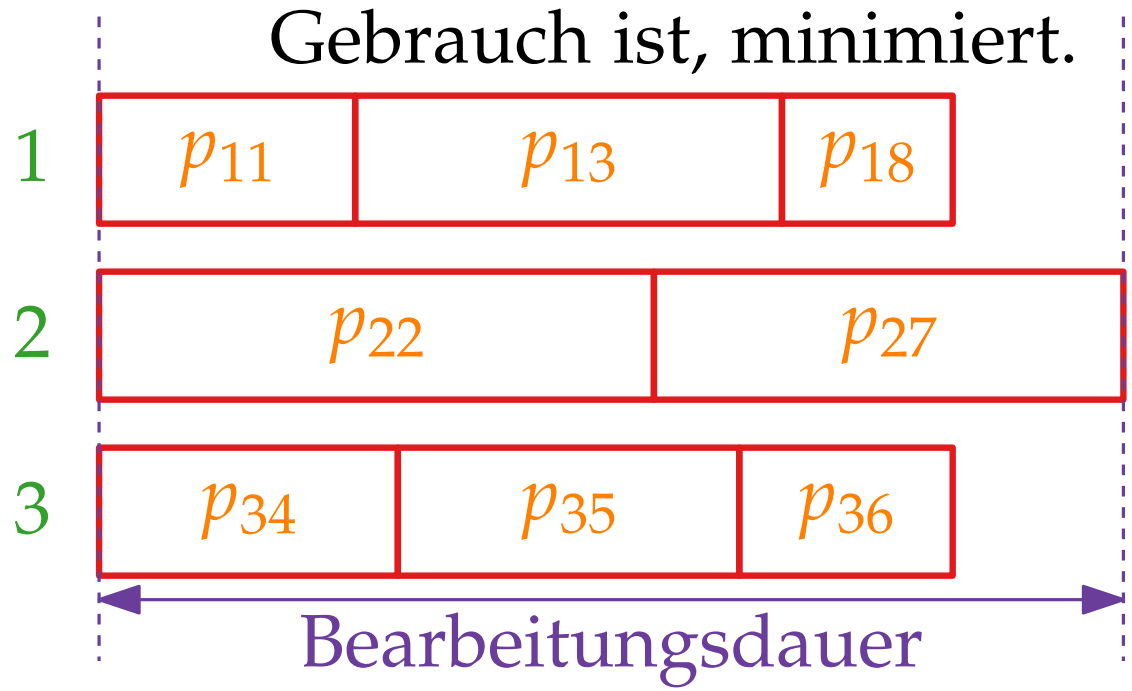
$$\mathcal{M} = \{M_1, M_2, M_3\}$$

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

Ablaufplanung auf parallelen Maschinen

Gegeben: Eine Menge $\mathcal{A}$ von **Aufträgen**,
eine Menge $\mathcal{M}$ von **Machinen**, und
für jedes $M_i \in \mathcal{M}$ und $A_j \in \mathcal{A}$ die
Bearbeitungszeit $p_{ij} \in \mathbb{N}^+$ von A_j auf M_i

Gesucht: Ein **Ablaufplan** $\sigma\colon \mathcal{A} \to \mathcal{M}$ der Aufträge auf den Maschinen, der die **Bearbeitungsdauer**, also die maximale Zeit, die eine Maschine in Gebrauch ist, minimiert.



$$\mathcal{A} = \{A_1, A_2, \ldots, A_8\}$$

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

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

Formulierung als ILP

minimize t

subject to

Formulierung als ILP

minimize t

subject to

$$x_{ij} \in \{0,1\}, \qquad M_i \in \mathcal{M}, A_j \in \mathcal{A}$$

Formulierung als ILP

$$
\begin{aligned}
&\textbf{minimize} && t && \\
&\textbf{subject to} && && A_j \in \mathcal{A} \\
& && && M_i \in \mathcal{M} \\
& && x_{ij} \in \{0,1\}, && M_i \in \mathcal{M}, A_j \in \mathcal{A}
\end{aligned}
$$

Formulierung als ILP

$$
\begin{aligned}
&\textbf{minimize} && t \\
&\textbf{subject to} && \sum_{M_i \in \mathcal{M}} x_{ij} = 1, && A_j \in \mathcal{A} \\
& && && M_i \in \mathcal{M} \\
& && x_{ij} \in \{0,1\}, && M_i \in \mathcal{M}, A_j \in \mathcal{A}
\end{aligned}
$$

Formulierung als ILP

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

Formulierung als ILP

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

Aufgabe: Zeige, dass das Integrality Gap unbeschränkt ist!

Formulierung als ILP

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

Aufgabe: Zeige, dass das Integrality Gap unbeschränkt ist!

Lösung:
m Machinen und ein Auftrag mit Bearbeitungszeit m

Formulierung als ILP

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

Aufgabe: Zeige, dass das Integrality Gap unbeschränkt ist!

Lösung:
m Machinen und ein Auftrag mit Bearbeitungszeit m
$\Rightarrow \mathrm{OPT} = m$ and $\mathrm{OPT}_{\mathrm{frac}} = 1$.

Parametrisiertes Stutzen

Verstärke das ILP $\rightarrow$ implizite (nicht-lineare) Nebenbedingungen:

Wenn $p_{ij} > t$, dann setze $x_{ij} = 0$.

Parametrisiertes Stutzen

Verstärke das ILP $\rightarrow$ implizite (nicht-lineare) Nebenbedingungen:

Wenn $p_{ij} > t$, dann setze $x_{ij} = 0$.

Führe einen neuen Parameter $T \in \mathbb{N}$ ein, um eine untere Schranke für OPT abzuschätzen.

Parametrisiertes Stutzen

Verstärke das ILP $\rightarrow$ implizite (nicht-lineare) Nebenbedingungen:

Wenn $p_{ij} > t$, dann setze $x_{ij} = 0$.

Führe einen neuen Parameter $T \in \mathbb{N}$ ein, um eine untere Schranke für OPT abzuschätzen.

Definiere $S_T := \{ (i,j) \colon M_i \in \mathcal{M},\ A_j \in \mathcal{A},\ p_{ij} \leq T \}$.

Parametrisiertes Stutzen

Verstärke das ILP $\rightarrow$ implizite (nicht-lineare) Nebenbedingungen:

Wenn $p_{ij} > t$, dann setze $x_{ij} = 0$.

Führe einen neuen Parameter $T \in \mathbb{N}$ ein, um eine untere Schranke für OPT abzuschätzen.

Definiere $S_T := \{ (i,j) \colon M_i \in \mathcal{M},\ A_j \in \mathcal{A},\ p_{ij} \leq T \}$.

Definiere die "gestutzte" Relaxierung LP(T):

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

Parametrisiertes Stutzen

Verstärke das ILP $\rightarrow$ implizite (nicht-lineare) Nebenbedingungen:

Wenn $p_{ij} > t$, dann setze $x_{ij} = 0$.

Führe einen neuen Parameter $T \in \mathbb{N}$ ein, um eine untere Schranke für OPT abzuschätzen.

Definiere $S_T := \{ (i,j) \colon M_i \in \mathcal{M},\ A_j \in \mathcal{A},\ p_{ij} \leq T \}$.

Definiere die "gestutzte" Relaxierung LP(T):

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

Parametrisiertes Stutzen

Verstärke das ILP $\rightarrow$ implizite (nicht-lineare) Nebenbedingungen:

Wenn $p_{ij} > t$, dann setze $x_{ij} = 0$.

Führe einen neuen Parameter $T \in \mathbb{N}$ ein, um eine untere Schranke für OPT abzuschätzen.

Definiere $S_T := \{ (i,j) \colon M_i \in \mathcal{M},\ A_j \in \mathcal{A},\ p_{ij} \leq T \}$.

Definiere die "gestutzte" Relaxierung LP(T):

$$
\begin{aligned}
&\textbf{minimize} && t \\
&\textbf{subject to} && \sum_{M_i \in \mathcal{M}} x_{ij} = 1, \qquad A_j \in \mathcal{A} \\
& && \sum_{A_j \in \mathcal{A}} x_{ij} p_{ij} \leq t, \qquad M_i \in \mathcal{M} \\
& && x_{ij} \cancel{\in \{0,1\}}, \geq 0 \quad \cancel{M_i \in \mathcal{M}, A_j \in \mathcal{A}} \quad (i,j) \in S_T
\end{aligned}
$$

Parametrisiertes Stutzen

Verstärke das ILP $\rightarrow$ implizite (nicht-lineare) Nebenbedingungen:

Wenn $p_{ij} > t$, dann setze $x_{ij} = 0$.

Führe einen neuen Parameter $T \in \mathbb{N}$ ein, um eine untere Schranke für OPT abzuschätzen.

Definiere $S_T := \{ (i,j) \colon M_i \in \mathcal{M},\ A_j \in \mathcal{A},\ p_{ij} \leq T \}$.

Definiere die "gestutzte" Relaxierung LP(T):

$$
\begin{aligned}
&\textbf{minimize} && t \\
&\textbf{subject to} && \sum_{(i,j) \in S_T \cancel{M_i \in \mathcal{M}}} x_{ij} = 1, \qquad A_j \in \mathcal{A} \\
& && \sum_{A_j \in \mathcal{A}} x_{ij} p_{ij} \leq t, \qquad M_i \in \mathcal{M} \\
& && x_{ij} \cancel{\in \{0,1\}}, \geq 0 \quad \cancel{M_i \in \mathcal{M}, A_j \in \mathcal{A}} \quad (i,j) \in S_T
\end{aligned}
$$

Parametrisiertes Stutzen

Verstärke das ILP $\rightarrow$ implizite (nicht-lineare) Nebenbedingungen:

Wenn $p_{ij} > t$, dann setze $x_{ij} = 0$.

Führe einen neuen Parameter $T \in \mathbb{N}$ ein, um eine untere Schranke für OPT abzuschätzen.

Definiere $S_T := \{ (i,j) \colon M_i \in \mathcal{M},\ A_j \in \mathcal{A},\ p_{ij} \leq T \}$.

Definiere die "gestutzte" Relaxierung LP(T):

$$
\begin{aligned}
&\textbf{minimize} && t \\
&\textbf{subject to} && \sum_{(i,j) \in S_T\ \cancel{M_i \in \mathcal{M}}} x_{ij} = 1, \qquad A_j \in \mathcal{A} \\
& && \sum_{(i,j) \in S_T\ \cancel{A_j \in \mathcal{A}}} x_{ij} p_{ij} \leq t, \qquad M_i \in \mathcal{M} \\
& && x_{ij} \cancel{\in \{0,1\}}, \geq 0 \quad \cancel{M_i \in \mathcal{M}, A_j \in \mathcal{A}} \quad (i,j) \in S_T
\end{aligned}
$$

Parametrisiertes Stutzen

Verstärke das ILP $\rightarrow$ implizite (nicht-lineare) Nebenbedingungen:

Wenn $p_{ij} > t$, dann setze $x_{ij} = 0$.

Führe einen neuen Parameter $T \in \mathbb{N}$ ein, um eine untere Schranke für OPT abzuschätzen.

Definiere $S_T := \{\, (i,j) \colon M_i \in \mathcal{M},\ A_j \in \mathcal{A},\ p_{ij} \leq T \,\}$.

Definiere die "gestutzte" Relaxierung LP(T):

$$
\begin{aligned}
&\textbf{minimize} && t \\
&\textbf{subject to} && \sum_{(i,j) \in S_T \cancel{M_i \in \mathcal{M}}} x_{ij} = 1, \qquad A_j \in \mathcal{A} \\
& && \sum_{(i,j) \in S_T \cancel{A_j \in \mathcal{A}}} x_{ij} p_{ij} \leq \cancel{t}\, T \quad M_i \in \mathcal{M} \\
& && x_{ij} \cancel{\in \{0,1\}}, \geq 0 \quad \cancel{M_i \in \mathcal{M}, A_j \in \mathcal{A}} \quad (i,j) \in S_T
\end{aligned}
$$

Parametrisiertes Stutzen

Verstärke das ILP $\rightarrow$ implizite (nicht-lineare) Nebenbedingungen:

Wenn $p_{ij} > t$, dann setze $x_{ij} = 0$.

Führe einen neuen Parameter $T \in \mathbb{N}$ ein, um eine untere Schranke für OPT abzuschätzen.

Definiere $S_T := \{\, (i,j) \colon M_i \in \mathcal{M},\ A_j \in \mathcal{A},\ p_{ij} \leq T \,\}$.

Definiere die "gestutzte" Relaxierung LP(T):

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

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

$$\sum_{(i,j) \in S_T\ \cancel{A_j \in \mathcal{A}}} x_{ij} p_{ij} \leq \cancel{t}\, T \quad M_i \in \mathcal{M}$$

$$x_{ij} \cancel{\in \{0,1\}}, \geq 0 \quad \cancel{M_i \in \mathcal{M}, A_j \in \mathcal{A}} \quad (i,j) \in S_T$$

Parametrisiertes Stutzen

Verstärke das ILP $\rightarrow$ implizite (nicht-lineare) Nebenbedingungen:

Wenn $p_{ij} > t$, dann setze $x_{ij} = 0$.

Führe einen neuen Parameter $T \in \mathbb{N}$ ein, um eine untere Schranke für OPT abzuschätzen.

Definiere $S_T := \{ (i,j) \colon M_i \in \mathcal{M},\ A_j \in \mathcal{A},\ p_{ij} \leq T \}$.

Definiere die "gestutzte" Relaxierung LP(T):

$$\boxed{\begin{array}{ll} \displaystyle\sum_{(i,j)\in S_T} x_{ij} = 1, & A_j \in \mathcal{A} \\ \displaystyle\sum_{(i,j)\in S_T} x_{ij} p_{ij} \leq T, & M_i \in \mathcal{M} \\ x_{ij} \geq 0, & (i,j) \in S_T \end{array}}$$

Parametrisiertes Stutzen

Verstärke das ILP $\rightarrow$ implizite (nicht-lineare) Nebenbedingungen:

Wenn $p_{ij} > t$, dann setze $x_{ij} = 0$.

Führe einen neuen Parameter $T \in \mathbb{N}$ ein, um eine untere Schranke für OPT abzuschätzen.

Definiere $S_T := \{ (i,j) \colon M_i \in \mathcal{M},\ A_j \in \mathcal{A},\ p_{ij} \leq T \}$.

Definiere die "gestutzte" Relaxierung LP(T):

$$\boxed{\begin{array}{ll} \displaystyle\sum_{(i,j)\in S_T} x_{ij} = 1, & A_j \in \mathcal{A} \\ \displaystyle\sum_{(i,j)\in S_T} x_{ij} p_{ij} \leq T, & M_i \in \mathcal{M} \\ x_{ij} \geq 0, & (i,j) \in S_T \end{array}}$$

LP(T) hat keine Zielfunktion; wir müssen nur entscheiden, ob eine zulässige Lösung existiert.

Parametrisiertes Stutzen

Verstärke das ILP $\rightarrow$ implizite (nicht-lineare) Nebenbedingungen:

Wenn $p_{ij} > t$, dann setze $x_{ij} = 0$.

Führe einen neuen Parameter $T \in \mathbb{N}$ ein, um eine untere Schranke für OPT abzuschätzen.

Definiere $S_T := \{ (i,j) \colon M_i \in \mathcal{M},\ A_j \in \mathcal{A},\ p_{ij} \leq T \}$.

Definiere die "gestutzte" Relaxierung LP(T):

$$\boxed{\begin{array}{ll} \sum\limits_{(i,j)\in S_T} x_{ij} = 1, & A_j \in \mathcal{A} \\ \sum\limits_{(i,j)\in S_T} x_{ij} p_{ij} \leq T, & M_i \in \mathcal{M} \\ x_{ij} \geq 0, & (i,j) \in S_T \end{array}}$$

LP(T) hat keine Zielfunktion; wir müssen nur entscheiden, ob eine zulässige Lösung existiert.

Aber warum gibt das LP ein gutes Integrality Gap?

Eigenschaften von Extrempunktlösungen

Benutze binäre Suche, um das kleinste T zu finden, so dass LP(T) eine zulässige Lösung hat.

$$\text{LP}(T)$$

$$\boxed{\begin{array}{ll} \displaystyle\sum_{(i,j)\in S_T} x_{ij} = 1, & A_j \in \mathcal{A} \\ \displaystyle\sum_{(i,j)\in S_T} x_{ij} p_{ij} \leq T, & M_i \in \mathcal{M} \\ x_{ij} \geq 0, & (i,j) \in S_T \end{array}}$$

Eigenschaften von Extrempunktlösungen

Benutze binäre Suche, um das kleinste T zu finden, so dass LP(T) eine zulässige Lösung hat. Sei T^* dieser Wert von T.

$$\text{LP}(T)$$

$$\boxed{\begin{array}{ll} \sum_{(i,j)\in S_T} x_{ij} = 1, & A_j \in \mathcal{A} \\ \sum_{(i,j)\in S_T} x_{ij} p_{ij} \leq T, & M_i \in \mathcal{M} \\ x_{ij} \geq 0, & (i,j) \in S_T \end{array}}$$

Eigenschaften von Extrempunktlösungen

Benutze binäre Suche, um das kleinste T zu finden, so dass LP(T) eine zulässige Lösung hat. Sei T^* dieser Wert von T.

Was sind die Schranken für unsere Suche?

LP(T)

$$\boxed{\begin{array}{ll} \sum_{(i,j)\in S_T} x_{ij} = 1, & A_j \in \mathcal{A} \\ \sum_{(i,j)\in S_T} x_{ij} p_{ij} \leq T, & M_i \in \mathcal{M} \\ x_{ij} \geq 0, & (i,j) \in S_T \end{array}}$$

Eigenschaften von Extrempunktlösungen

Benutze binäre Suche, um das kleinste T zu finden, so dass LP(T) eine zulässige Lösung hat. Sei T^* dieser Wert von T.

Was sind die Schranken für unsere Suche?

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

LP(T)

$$\boxed{\begin{array}{ll} \sum_{(i,j)\in S_T} x_{ij} = 1, & A_j \in \mathcal{A} \\ \sum_{(i,j)\in S_T} x_{ij} p_{ij} \leq T, & M_i \in \mathcal{M} \\ x_{ij} \geq 0, & (i,j) \in S_T \end{array}}$$

Eigenschaften von Extrempunktlösungen

Benutze binäre Suche, um das kleinste T zu finden, so dass LP(T) eine zulässige Lösung hat. Sei T^* dieser Wert von T.

Was sind die Schranken für unsere Suche?

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

Idee: Runde eine Extrempunktlösung von LP(T^*) zu einem Ablaufplan mit Produktionsdauer $\leq 2T^*$.

LP(T)

$$\boxed{\begin{array}{ll} \sum_{(i,j)\in S_T} x_{ij} = 1, & A_j \in \mathcal{A} \\ \sum_{(i,j)\in S_T} x_{ij} p_{ij} \leq T, & M_i \in \mathcal{M} \\ x_{ij} \geq 0, & (i,j) \in S_T \end{array}}$$

Eigenschaften von Extrempunktlösungen

Benutze binäre Suche, um das kleinste T zu finden, so dass LP(T) eine zulässige Lösung hat. Sei T^* dieser Wert von T.

Was sind die Schranken für unsere Suche?

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

Idee: Runde eine Extrempunktlösung von LP(T^*) zu einem Ablaufplan mit Produktionsdauer $\leq 2T^*$.

LP(T)

$$\sum_{(i,j)\in S_T} x_{ij} = 1, \qquad A_j \in \mathcal{A}$$

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

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

Lemma 1.
Jede Extrempunktlösung für LP(T) hat höchstens $|\mathcal{M}| + |\mathcal{A}|$ pos. Variablen.

Eigenschaften von Extrempunktlösungen

Benutze binäre Suche, um das kleinste T zu finden, so dass LP(T) eine zulässige Lösung hat. Sei T^* dieser Wert von T.

Was sind die Schranken für unsere Suche?

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

Idee: Runde eine Extrempunktlösung von LP(T^*) zu einem Ablaufplan mit Produktionsdauer $\leq 2T^*$.

LP(T)

$$\sum_{(i,j)\in S_T} x_{ij} = 1, \qquad A_j \in \mathcal{A}$$

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

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

Lemma 1.
Jede Extrempunktlösung für LP(T) hat höchstens $|\mathcal{M}| + |\mathcal{A}|$ pos. Variablen.

Lemma 2.
Jede Extrempunktlsg. für LP(T) belegt min. $|\mathcal{A}| - |\mathcal{M}|$ Aufträge ganzzahlig.

Eigenschaften von Extrempunktlösungen

$$\boxed{\begin{array}{ll} \sum_{(i,j)\in S_T} x_{ij} = 1, & A_j \in \mathcal{A} \\ \sum_{(i,j)\in S_T} x_{ij} p_{ij} \leq T, & M_i \in \mathcal{M} \\ x_{ij} \geq 0, & (i,j) \in S_T \end{array}}$$

Lemma 1.
Jede Extrempunktlösung für LP(T) hat höchstens $|\mathcal{M}| + |\mathcal{A}|$ positive Variablen.

Eigenschaften von Extrempunktlösungen

$$\boxed{\begin{array}{ll} \sum_{(i,j)\in S_T} x_{ij} = 1, & A_j \in \mathcal{A} \\ \sum_{(i,j)\in S_T} x_{ij} p_{ij} \leq T, & M_i \in \mathcal{M} \\ x_{ij} \geq 0, & (i,j) \in S_T \end{array}}$$

Lemma 1.
Jede Extrempunktlösung für LP(T) hat höchstens $|\mathcal{M}| + |\mathcal{A}|$ positive Variablen.

Beweis. $L(T)$: $|S_T|$ Variablen

Eigenschaften von Extrempunktlösungen

$$\boxed{\begin{array}{ll} \sum_{(i,j)\in S_T} x_{ij} = 1, & A_j \in \mathcal{A} \\ \sum_{(i,j)\in S_T} x_{ij} p_{ij} \leq T, & M_i \in \mathcal{M} \\ x_{ij} \geq 0, & (i,j) \in S_T \end{array}}$$

Lemma 1.
Jede Extrempunktlösung für LP(T) hat höchstens $|\mathcal{M}| + |\mathcal{A}|$ positive Variablen.

Beweis. $L(T)$: $|S_T|$ Variablen
Extrempunktlsg.: $|S_T|$ Ungleichungen genau erfüllt

Eigenschaften von Extrempunktlösungen

$$\sum_{(i,j)\in S_T} x_{ij} = 1, \qquad A_j \in \mathcal{A}$$

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

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

Lemma 1.
Jede Extrempunktlösung für LP(T) hat höchstens $|\mathcal{M}| + |\mathcal{A}|$ positive Variablen.

Beweis. $L(T)$: $|S_T|$ Variablen
Extrempunktlsg.: $|S_T|$ Ungleichungen genau erfüllt
max. $|\mathcal{A}|$

Eigenschaften von Extrempunktlösungen

$$\sum_{(i,j)\in S_T} x_{ij} = 1, \qquad A_j \in \mathcal{A}$$

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

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

Lemma 1.
Jede Extrempunktlösung für LP(T) hat höchstens $|\mathcal{M}| + |\mathcal{A}|$ positive Variablen.

Beweis. $L(T)$: $|S_T|$ Variablen
Extrempunktlsg.: $|S_T|$ Ungleichungen genau erfüllt

→ max. $|\mathcal{A}|$

→ max. $|\mathcal{M}|$

Eigenschaften von Extrempunktlösungen

$$\sum_{(i,j)\in S_T} x_{ij} = 1, \qquad A_j \in \mathcal{A}$$

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

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

Lemma 1.
Jede Extrempunktlösung für LP(T) hat höchstens $|\mathcal{M}| + |\mathcal{A}|$ positive Variablen.

Beweis. $L(T)$: $|S_T|$ Variablen
Extrempunktlsg.: $|S_T|$ Ungleichungen genau erfüllt

max. $|\mathcal{A}|$

max. $|\mathcal{M}|$

$\Rightarrow$ min. $|S_T| - |\mathcal{A}| - |\mathcal{M}|$

Eigenschaften von Extrempunktlösungen

$$\sum_{(i,j)\in S_T} x_{ij} = 1, \qquad A_j \in \mathcal{A}$$

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

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

Lemma 1.
Jede Extrempunktlösung für LP(T) hat höchstens $|\mathcal{M}| + |\mathcal{A}|$ positive Variablen.

Beweis. $L(T)$: $|S_T|$ Variablen
Extrempunktlsg.: $|S_T|$ Ungleichungen genau erfüllt
max. $|\mathcal{A}|$
max. $|\mathcal{M}|$
$\Rightarrow$ min. $|S_T| - |\mathcal{A}| - |\mathcal{M}|$
$\Rightarrow$ max. $|\mathcal{M}| + |\mathcal{A}|$ nicht erfüllt $\square$

Eigenschaften von Extrempunktlösungen

$$\boxed{\begin{array}{ll} \sum_{(i,j)\in S_T} x_{ij} = 1, & A_j \in \mathcal{A} \\ \sum_{(i,j)\in S_T} x_{ij} p_{ij} \leq T, & M_i \in \mathcal{M} \\ x_{ij} \geq 0, & (i,j) \in S_T \end{array}}$$

Lemma 2.
Jede Extrempunktlösung für LP(T) belegt min. $|\mathcal{A}|-|\mathcal{M}|$ Aufträge ganzzahlig.

Eigenschaften von Extrempunktlösungen

$$\boxed{\begin{array}{ll} \sum_{(i,j)\in S_T} x_{ij} = 1, & A_j \in \mathcal{A} \\ \sum_{(i,j)\in S_T} x_{ij} p_{ij} \leq T, & M_i \in \mathcal{M} \\ x_{ij} \geq 0, & (i,j) \in S_T \end{array}}$$

Lemma 2.
Jede Extrempunktlösung für LP(T) belegt min. $|\mathcal{A}|-|\mathcal{M}|$ Aufträge ganzzahlig.

Beweis. Sei x Extrempunktlösung von $L(T)$.

Eigenschaften von Extrempunktlösungen

$$\boxed{\begin{array}{ll} \sum_{(i,j)\in S_T} x_{ij} = 1, & A_j \in \mathcal{A} \\ \sum_{(i,j)\in S_T} x_{ij} p_{ij} \leq T, & M_i \in \mathcal{M} \\ x_{ij} \geq 0, & (i,j) \in S_T \end{array}}$$

Lemma 2.
Jede Extrempunktlösung für LP(T) belegt min. $|\mathcal{A}|-|\mathcal{M}|$ Aufträge ganzzahlig.

Beweis. Sei x Extrempunktlösung von $L(T)$.
Seien α Aufträge ganzzahlig und β fraktional in x.

Eigenschaften von Extrempunktlösungen

$$\boxed{\begin{array}{ll} \sum_{(i,j)\in S_T} x_{ij} = 1, & A_j \in \mathcal{A} \\ \sum_{(i,j)\in S_T} x_{ij} p_{ij} \leq T, & M_i \in \mathcal{M} \\ x_{ij} \geq 0, & (i,j) \in S_T \end{array}}$$

Lemma 2.
Jede Extrempunktlösung für LP(T) belegt min. $|\mathcal{A}|-|\mathcal{M}|$ Aufträge ganzzahlig.

Beweis. Sei x Extrempunktlösung von $L(T)$.
Seien α Aufträge ganzzahlig und β fraktional in x.
$\Rightarrow \alpha + \beta = |\mathcal{A}|$

Eigenschaften von Extrempunktlösungen

$$\boxed{\begin{array}{ll} \sum\limits_{(i,j)\in S_T} x_{ij} = 1, & A_j \in \mathcal{A} \\ \sum\limits_{(i,j)\in S_T} x_{ij} p_{ij} \leq T, & M_i \in \mathcal{M} \\ x_{ij} \geq 0, & (i,j) \in S_T \end{array}}$$

Lemma 2.
Jede Extrempunktlösung für LP(T) belegt min. $|\mathcal{A}|-|\mathcal{M}|$ Aufträge ganzzahlig.

Beweis. Sei x Extrempunktlösung von $L(T)$.
Seien α Aufträge ganzzahlig und β fraktional in x.
$\Rightarrow \alpha + \beta = |\mathcal{A}|$
Fraktionale Aufträge: min. 2 Maschinen

Eigenschaften von Extrempunktlösungen

$$\boxed{\begin{array}{ll} \sum\limits_{(i,j)\in S_T} x_{ij} = 1, & A_j \in \mathcal{A} \\ \sum\limits_{(i,j)\in S_T} x_{ij} p_{ij} \leq T, & M_i \in \mathcal{M} \\ x_{ij} \geq 0, & (i,j) \in S_T \end{array}}$$

Lemma 2.
Jede Extrempunktlösung für LP(T) belegt min. $|\mathcal{A}|-|\mathcal{M}|$ Aufträge ganzzahlig.

Beweis. Sei x Extrempunktlösung von $L(T)$.
Seien α Aufträge ganzzahlig und β fraktional in x.
$\Rightarrow \alpha + \beta = |\mathcal{A}|$
Fraktionale Aufträge: min. 2 Maschinen
$\Rightarrow$ min. 2 Variablen > 0

Eigenschaften von Extrempunktlösungen

$$\boxed{\begin{array}{ll} \sum\limits_{(i,j)\in S_T} x_{ij} = 1, & A_j \in \mathcal{A} \\ \sum\limits_{(i,j)\in S_T} x_{ij} p_{ij} \leq T, & M_i \in \mathcal{M} \\ x_{ij} \geq 0, & (i,j) \in S_T \end{array}}$$

Lemma 2.
Jede Extrempunktlösung für LP(T) belegt min. $|\mathcal{A}|-|\mathcal{M}|$ Aufträge ganzzahlig.

Beweis. Sei x Extrempunktlösung von $L(T)$.
Seien α Aufträge ganzzahlig und β fraktional in x.
$\Rightarrow \alpha + \beta = |\mathcal{A}|$
Fraktionale Aufträge: min. 2 Maschinen
$\Rightarrow$ min. 2 Variablen > 0
$\Rightarrow \alpha + 2\beta \leq |\mathcal{A}| + |\mathcal{M}|$ (Lemma 1)

Eigenschaften von Extrempunktlösungen

$$\boxed{\begin{array}{ll} \sum\limits_{(i,j)\in S_T} x_{ij} = 1, & A_j \in \mathcal{A} \\ \sum\limits_{(i,j)\in S_T} x_{ij} p_{ij} \leq T, & M_i \in \mathcal{M} \\ x_{ij} \geq 0, & (i,j) \in S_T \end{array}}$$

Lemma 2.
Jede Extrempunktlösung für LP(T) belegt min. $|\mathcal{A}|-|\mathcal{M}|$ Aufträge ganzzahlig.

Beweis. Sei x Extrempunktlösung von $L(T)$.
Seien α Aufträge ganzzahlig und β fraktional in x.
$\Rightarrow \alpha + \beta = |\mathcal{A}|$
Fraktionale Aufträge: min. 2 Maschinen
$\Rightarrow$ min. 2 Variablen > 0
$\Rightarrow \alpha + 2\beta \leq |\mathcal{A}| + |\mathcal{M}|$ (Lemma 1)
$\Rightarrow \beta \leq |\mathcal{M}|$

Eigenschaften von Extrempunktlösungen

$$\boxed{\begin{array}{ll} \sum\limits_{(i,j)\in S_T} x_{ij} = 1, & A_j \in \mathcal{A} \\ \sum\limits_{(i,j)\in S_T} x_{ij} p_{ij} \leq T, & M_i \in \mathcal{M} \\ x_{ij} \geq 0, & (i,j) \in S_T \end{array}}$$

Lemma 2.
Jede Extrempunktlösung für LP(T) belegt min. $|\mathcal{A}|-|\mathcal{M}|$ Aufträge ganzzahlig.

Beweis. Sei x Extrempunktlösung von $L(T)$.
Seien α Aufträge ganzzahlig und β fraktional in x.
$\Rightarrow \alpha + \beta = |\mathcal{A}|$
Fraktionale Aufträge: min. 2 Maschinen
$\Rightarrow$ min. 2 Variablen > 0
$\Rightarrow \alpha + 2\beta \leq |\mathcal{A}| + |\mathcal{M}|$ (Lemma 1)
$\Rightarrow \beta \leq |\mathcal{M}| \quad \Rightarrow \alpha \geq |\mathcal{A}| - |\mathcal{M}|$ $\square$

Extrempunktlösungen von LP(T)

Definiere: Bipartiter Graph $G = (\mathcal{A} \cup \mathcal{M}, E)$
mit $(i, j) \in E \Leftrightarrow x_{ij} \neq 0$.

Extrempunktlösungen von LP(T)

Definiere: Bipartiter Graph $G = (\mathcal{A} \cup \mathcal{M}, E)$
mit $(i, j) \in E \Leftrightarrow x_{ij} \neq 0$.

Aufträge können *ganzzahlig* oder *fraktional* zugewiesen werden.

Extrempunktlösungen von LP(T)

Definiere: Bipartiter Graph $G = (\mathcal{A} \cup \mathcal{M}, E)$
mit $(i, j) \in E \Leftrightarrow x_{ij} \neq 0$.

Aufträge können *ganzzahlig* oder *fraktional* zugewiesen werden. $(\exists M_i \in \mathcal{M} : 0 < x_{ij} < 1)$

Extrempunktlösungen von LP(T)

Definiere: Bipartiter Graph $G = (\mathcal{A} \cup \mathcal{M}, E)$
mit $(i, j) \in E \Leftrightarrow x_{ij} \neq 0$.

Aufträge können *ganzzahlig* oder *fraktional* zugewiesen werden. $(\exists M_i \in \mathcal{M} : 0 < x_{ij} < 1)$

Sei $F \subseteq \mathcal{A}$ die Menge der fraktional zugewiesenen Aufträge.

Extrempunktlösungen von LP(T)

Definiere: Bipartiter Graph $G = (\mathcal{A} \cup \mathcal{M}, E)$
mit $(i, j) \in E \Leftrightarrow x_{ij} \neq 0$.

Aufträge können *ganzzahlig* oder *fraktional* zugewiesen werden. $(\exists M_i \in \mathcal{M} : 0 < x_{ij} < 1)$

Sei $F \subseteq \mathcal{A}$ die Menge der fraktional zugewiesenen Aufträge.
Sei $H := G[F \cup \mathcal{M}]$.

Extrempunktlösungen von LP(T)

Definiere: Bipartiter Graph $G = (\mathcal{A} \cup \mathcal{M}, E)$ mit $(i,j) \in E \Leftrightarrow x_{ij} \neq 0$.

Aufträge können *ganzzahlig* oder *fraktional* zugewiesen werden. $(\exists M_i \in \mathcal{M} : 0 < x_{ij} < 1)$

Sei $F \subseteq \mathcal{A}$ die Menge der fraktional zugewiesenen Aufträge.
Sei $H := G[F \cup \mathcal{M}]$.

Beobachte: (i,j) ist eine Kante in $H \Leftrightarrow 0 < x_{ij} < 1$

Extrempunktlösungen von LP(T)

Definiere: Bipartiter Graph $G = (\mathcal{A} \cup \mathcal{M}, E)$ mit $(i,j) \in E \Leftrightarrow x_{ij} \neq 0$.

Aufträge können *ganzzahlig* oder *fraktional* zugewiesen werden. $(\exists M_i \in \mathcal{M} : 0 < x_{ij} < 1)$

Sei $F \subseteq \mathcal{A}$ die Menge der fraktional zugewiesenen Aufträge. Sei $H := G[F \cup \mathcal{M}]$.

Beobachte: (i,j) ist eine Kante in $H \Leftrightarrow 0 < x_{ij} < 1$

Ein Matching in H wird F-*perfekt* genannt, wenn es jeden Knoten in F paart.

Extrempunktlösungen von LP(T)

Definiere: Bipartiter Graph $G = (\mathcal{A} \cup \mathcal{M}, E)$ mit $(i,j) \in E \Leftrightarrow x_{ij} \neq 0$.

Aufträge können *ganzzahlig* oder *fraktional* zugewiesen werden. $(\exists M_i \in \mathcal{M} : 0 < x_{ij} < 1)$

Sei $F \subseteq \mathcal{A}$ die Menge der fraktional zugewiesenen Aufträge. Sei $H := G[F \cup \mathcal{M}]$.

Beobachte: (i,j) ist eine Kante in $H \Leftrightarrow 0 < x_{ij} < 1$

Ein Matching in H wird F-*perfekt* genannt, wenn es jeden Knoten in F paart.

Hauptschritt: Zeige, dass H immer ein F-perfektes Matching hat.

Extrempunktlösungen von LP(T)

Definiere: Bipartiter Graph $G = (\mathcal{A} \cup \mathcal{M}, E)$ mit $(i,j) \in E \Leftrightarrow x_{ij} \neq 0$.

Aufträge können *ganzzahlig* oder *fraktional* zugewiesen werden. $(\exists M_i \in \mathcal{M} : 0 < x_{ij} < 1)$

Sei $F \subseteq \mathcal{A}$ die Menge der fraktional zugewiesenen Aufträge. Sei $H := G[F \cup \mathcal{M}]$.

Beobachte: (i,j) ist eine Kante in $H \Leftrightarrow 0 < x_{ij} < 1$

Ein Matching in H wird F-*perfekt* genannt, wenn es jeden Knoten in F paart.

Hauptschritt: Zeige, dass H immer ein F-perfektes Matching hat.

Warum ist das nützlich ...?

Algorithmus

Weise Auftrag A_j zu der Maschine M_i zu, die p_{ij} minimiert.

Algorithmus

Weise Auftrag A_j zu der Maschine M_i zu, die p_{ij} minimiert. Sei τ die Bearbeitungsdauer dieses Ablaufplans.

Algorithmus

Weise Auftrag A_j zu der Maschine M_i zu, die p_{ij} minimiert. Sei τ die Bearbeitungsdauer dieses Ablaufplans. Verwende binäre Suche im Interval $[\frac{\tau}{|\mathcal{M}|}, \tau]$, um den kleinsten Wert von $T \in \mathbb{Z}^+$, für den LP(T) eine zulässige Lösung hat, zu finden. Sei T^* dieser Wert.

Algorithmus

Weise Auftrag A_j zu der Maschine M_i zu, die p_{ij} minimiert. Sei τ die Bearbeitungsdauer dieses Ablaufplans.

Verwende binäre Suche im Interval $[\frac{\tau}{|\mathcal{M}|}, \tau]$, um den kleinsten Wert von $T \in \mathbb{Z}^+$, für den LP(T) eine zulässige Lösung hat, zu finden. Sei T^* dieser Wert.

Finde eine Extrempunktlösung x für LP(T^*).

Algorithmus

Weise Auftrag A_j zu der Maschine M_i zu, die p_{ij} minimiert. Sei τ die Bearbeitungsdauer dieses Ablaufplans.

Verwende binäre Suche im Interval $[\frac{\tau}{|\mathcal{M}|}, \tau]$, um den kleinsten Wert von $T \in \mathbb{Z}^+$, für den LP(T) eine zulässige Lösung hat, zu finden. Sei T^* dieser Wert.
Finde eine Extrempunktlösung x für LP(T^*).

Weise alle ganzzahligen Aufträge wie in x zu.

Algorithmus

Weise Auftrag A_j zu der Maschine M_i zu, die p_{ij} minimiert. Sei τ die Bearbeitungsdauer dieses Ablaufplans.

Verwende binäre Suche im Interval $[\frac{\tau}{|\mathcal{M}|}, \tau]$, um den kleinsten Wert von $T \in \mathbb{Z}^+$, für den LP(T) eine zulässige Lösung hat, zu finden. Sei T^* dieser Wert.
Finde eine Extrempunktlösung x für LP(T^*).

Weise alle ganzzahligen Aufträge wie in x zu.

Erstelle den Graph H und finde ein F-perfektes Matching P (siehe Lemma 4 später).

Algorithmus

Weise Auftrag A_j zu der Maschine M_i zu, die p_{ij} minimiert. Sei τ die Bearbeitungsdauer dieses Ablaufplans.

Verwende binäre Suche im Interval $[\frac{\tau}{|\mathcal{M}|}, \tau]$, um den kleinsten Wert von $T \in \mathbb{Z}^+$, für den LP(T) eine zulässige Lösung hat, zu finden. Sei T^* dieser Wert.
Finde eine Extrempunktlösung x für LP(T^*).

Weise alle ganzzahligen Aufträge wie in x zu.

Erstelle den Graph H und finde ein F-perfektes Matching P (siehe Lemma 4 später).

Weise die fraktionalen Aufträge entsprechend P zu.

Algorithmus

Weise Auftrag A_j zu der Maschine M_i zu, die p_{ij} minimiert. Sei τ die Bearbeitungsdauer dieses Ablaufplans.

Verwende binäre Suche im Interval $[\frac{\tau}{|\mathcal{M}|}, \tau]$, um den kleinsten Wert von $T \in \mathbb{Z}^+$, für den LP(T) eine zulässige Lösung hat, zu finden. Sei T^* dieser Wert.
Finde eine Extrempunktlösung x für LP(T^*).

Weise alle ganzzahligen Aufträge wie in x zu.

Erstelle den Graph H und finde ein F-perfektes Matching P (siehe Lemma 4 später).

Weise die fraktionalen Aufträge entsprechend P zu.

Satz. Dieser Algorithmus ist eine Faktor-2-Approximation (vorausgesetzt wir haben ein F-perfektes Matching).

Approximationsfaktor

$$\boxed{\begin{array}{lll} \sum\limits_{(i,j)\in S_{T^*}} x_{ij} = 1, & & A_j \in \mathcal{A} \\ \sum\limits_{(i,j)\in S_{T^*}} x_{ij} p_{ij} \leq T^*, & & M_i \in \mathcal{M} \\ x_{ij} \geq 0, & & (i,j) \in S \end{array}}$$

Satz. Dieser Algorithmus ist eine Faktor-2-Approximation (vorausgesetzt wir haben ein F-perfektes Matching).

Beweis. $T^* \leq \mathrm{OPT}$

Approximationsfaktor

$$\boxed{\begin{array}{lll} \sum\limits_{(i,j)\in S_{T^*}} x_{ij} = 1, & & A_j \in \mathcal{A} \\ \sum\limits_{(i,j)\in S_{T^*}} x_{ij} p_{ij} \leq T^*, & & M_i \in \mathcal{M} \\ x_{ij} \geq 0, & & (i,j) \in S \end{array}}$$

Satz. Dieser Algorithmus ist eine Faktor-2-Approximation (vorausgesetzt wir haben ein F-perfektes Matching).

Beweis. $T^* \leq \mathrm{OPT}$

Sei x Extrempunktlösung für $LP(T^*)$

Approximationsfaktor

$$\sum_{(i,j)\in S_{T^*}} x_{ij} = 1, \qquad A_j \in \mathcal{A}$$

$$\sum_{(i,j)\in S_{T^*}} x_{ij} p_{ij} \leq T^*, \qquad M_i \in \mathcal{M}$$

$$x_{ij} \geq 0, \qquad (i,j) \in S$$

Satz. Dieser Algorithmus ist eine Faktor-2-Approximation (vorausgesetzt wir haben ein F-perfektes Matching).

Beweis. $T^* \leq \mathrm{OPT}$

Sei x Extrempunktlösung für $LP(T^*)$

Fraktionale Lösung: Bearbeitungsdauer $\leq T^*$.

Approximationsfaktor

$$\sum_{(i,j)\in S_{T^*}} x_{ij} = 1, \qquad A_j \in \mathcal{A}$$

$$\sum_{(i,j)\in S_{T^*}} x_{ij} p_{ij} \leq T^*, \qquad M_i \in \mathcal{M}$$

$$x_{ij} \geq 0, \qquad (i,j) \in S$$

Satz. Dieser Algorithmus ist eine Faktor-2-Approximation (vorausgesetzt wir haben ein F-perfektes Matching).

Beweis. $T^* \leq \mathrm{OPT}$

Sei x Extrempunktlösung für $LP(T^*)$

Fraktionale Lösung: Bearbeitungsdauer $\leq T^*$.

$\Rightarrow$ Einschränkung auf ganzzahlige Aufträge hat Bearbeitungsdauer $\leq T^*$

Approximationsfaktor

$$\sum_{(i,j)\in S_{T^*}} x_{ij} = 1, \qquad A_j \in \mathcal{A}$$

$$\sum_{(i,j)\in S_{T^*}} x_{ij} p_{ij} \leq T^*, \qquad M_i \in \mathcal{M}$$

$$x_{ij} \geq 0, \qquad (i,j) \in S$$

Satz. Dieser Algorithmus ist eine Faktor-2-Approximation (vorausgesetzt wir haben ein F-perfektes Matching).

Beweis. $T^* \leq \mathrm{OPT}$

Sei x Extrempunktlösung für $LP(T^*)$

Fraktionale Lösung: Bearbeitungsdauer $\leq T^*$.

$\Rightarrow$ Einschränkung auf ganzzahlige Aufträge hat Bearbeitungsdauer $\leq T^*$

Für jede Kante $(i,j) \in S_{T^*}$: $p_{ij} \leq T^*$

Approximationsfaktor

$$\sum_{(i,j)\in S_{T^*}} x_{ij} = 1, \qquad A_j \in \mathcal{A}$$

$$\sum_{(i,j)\in S_{T^*}} x_{ij} p_{ij} \leq T^*, \qquad M_i \in \mathcal{M}$$

$$x_{ij} \geq 0, \qquad (i,j) \in S$$

Satz. Dieser Algorithmus ist eine Faktor-2-Approximation (vorausgesetzt wir haben ein F-perfektes Matching).

Beweis. $T^* \leq \mathrm{OPT}$

Sei x Extrempunktlösung für $LP(T^*)$

Fraktionale Lösung: Bearbeitungsdauer $\leq T^*$.

$\Rightarrow$ Einschränkung auf ganzzahlige Aufträge hat Bearbeitungsdauer $\leq T^*$

Für jede Kante $(i,j) \in S_{T^*}$: $p_{ij} \leq T^*$

Matching: max. 1 extra Auftrag pro Maschine

Approximationsfaktor

$$\sum_{(i,j)\in S_{T^*}} x_{ij} = 1, \qquad A_j \in \mathcal{A}$$

$$\sum_{(i,j)\in S_{T^*}} x_{ij} p_{ij} \leq T^*, \qquad M_i \in \mathcal{M}$$

$$x_{ij} \geq 0, \qquad (i,j) \in S$$

Satz. Dieser Algorithmus ist eine Faktor-2-Approximation (vorausgesetzt wir haben ein F-perfektes Matching).

Beweis. $T^* \leq \mathrm{OPT}$

Sei x Extrempunktlösung für $LP(T^*)$

Fraktionale Lösung: Bearbeitungsdauer $\leq T^*$.

$\Rightarrow$ Einschränkung auf ganzzahlige Aufträge hat Bearbeitungsdauer $\leq T^*$

Für jede Kante $(i,j) \in S_{T^*}$: $p_{ij} \leq T^*$

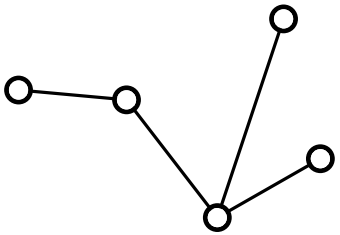
Matching: max. 1 extra Auftrag pro Maschine

$\Rightarrow$ gesamte Bearbeitungsdauer $\leq 2T^* \leq 2\mathrm{OPT}$ $\square$

Pseudo-Bäume and -Wälder

Pseudo-Baum: Zusammenhängender Graph mit Knotenmenge V wird und höchstens $|V|$ Kanten.

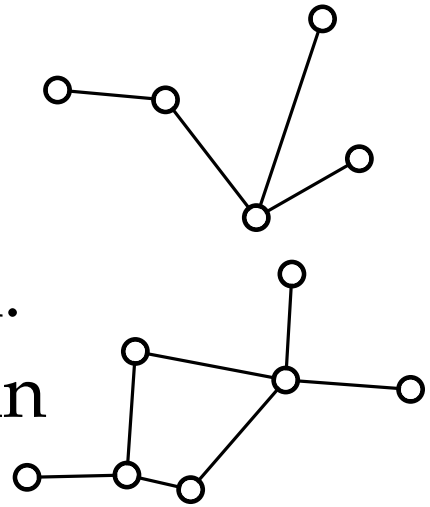
Pseudo-Bäume and -Wälder



Pseudo-Baum: Zusammenhängender Graph mit Knotenmenge V wird und höchstens $|V|$ Kanten.

Ein Pseudo-Baum ist entweder ein Baum

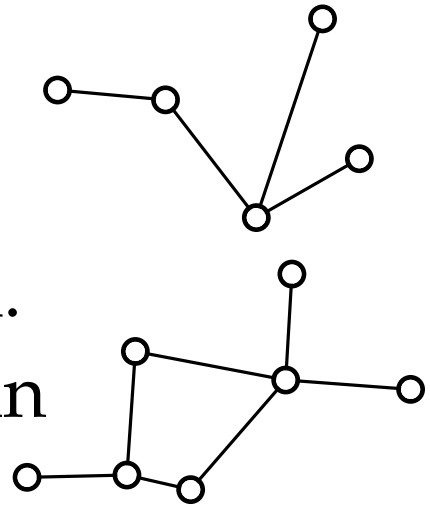
Pseudo-Bäume and -Wälder



Pseudo-Baum: Zusammenhängender Graph mit Knotenmenge V wird und höchstens $|V|$ Kanten.

Ein Pseudo-Baum ist entweder ein Baum oder ein Baum mit einer zusätzlichlichen Kante.

Pseudo-Bäume and -Wälder

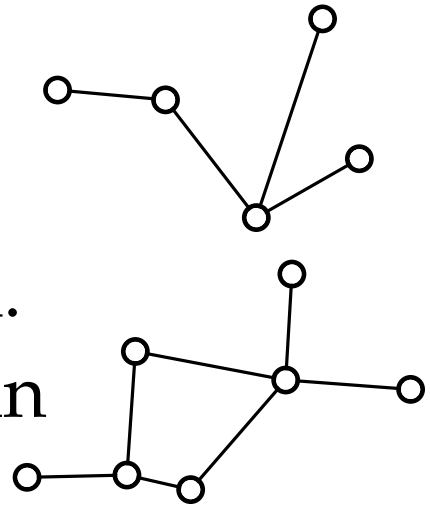


Pseudo-Baum: Zusammenhängender Graph mit Knotenmenge V wird und höchstens $|V|$ Kanten.

Ein Pseudo-Baum ist entweder ein Baum oder ein Baum mit einer zusätzlichlichen Kante.

Pseudo-Wald: Sammlung von disjunkten Pseudo-Bäumen

Pseudo-Bäume and -Wälder



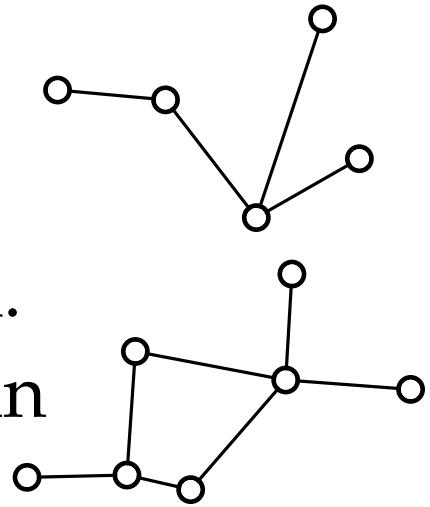
Pseudo-Baum: Zusammenhängender Graph mit Knotenmenge V wird und höchstens $|V|$ Kanten.

Ein Pseudo-Baum ist entweder ein Baum oder ein Baum mit einer zusätzlichlichen Kante.

Pseudo-Wald: Sammlung von disjunkten Pseudo-Bäumen

Lemma 3.
Der bipartite Graph $G = (\mathcal{A} \cup \mathcal{M}, E)$ ist ein Pseudo-Wald.

Pseudo-Bäume and -Wälder



Pseudo-Baum: Zusammenhängender Graph mit Knotenmenge V wird und höchstens $|V|$ Kanten.

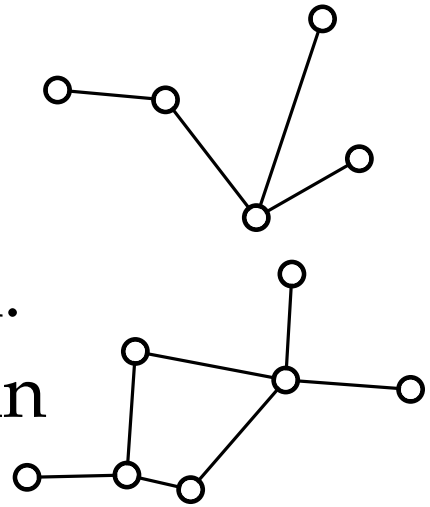
Ein Pseudo-Baum ist entweder ein Baum oder ein Baum mit einer zusätzlichlichen Kante.

Pseudo-Wald: Sammlung von disjunkten Pseudo-Bäumen

Lemma 3.
Der bipartite Graph $G = (\mathcal{A} \cup \mathcal{M}, E)$ ist ein Pseudo-Wald.

Extrempunkt-Lösungen haben $\leq |\mathcal{M}| + |\mathcal{A}|$ Variablen > 0 (L1).

Pseudo-Bäume and -Wälder



Pseudo-Baum: Zusammenhängender Graph mit Knotenmenge V wird und höchstens $|V|$ Kanten.

Ein Pseudo-Baum ist entweder ein Baum oder ein Baum mit einer zusätzlichlichen Kante.

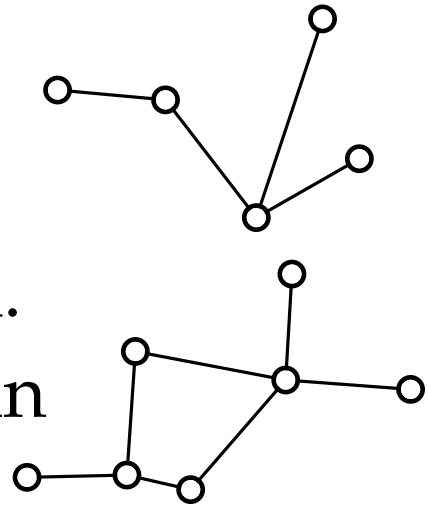
Pseudo-Wald: Sammlung von disjunkten Pseudo-Bäumen

Lemma 3.
Der bipartite Graph $G = (\mathcal{A} \cup \mathcal{M}, E)$ ist ein Pseudo-Wald.

Extrempunkt-Lösungen haben $\leq |\mathcal{M}| + |\mathcal{A}|$ Variablen > 0 (L1).

Jede Komponente von G entspricht einer Extrempunkt-Lösung.

Pseudo-Bäume and -Wälder



Pseudo-Baum: Zusammenhängender Graph mit Knotenmenge V wird und höchstens $|V|$ Kanten.

Ein Pseudo-Baum ist entweder ein Baum oder ein Baum mit einer zusätzlichlichen Kante.

Pseudo-Wald: Sammlung von disjunkten Pseudo-Bäumen

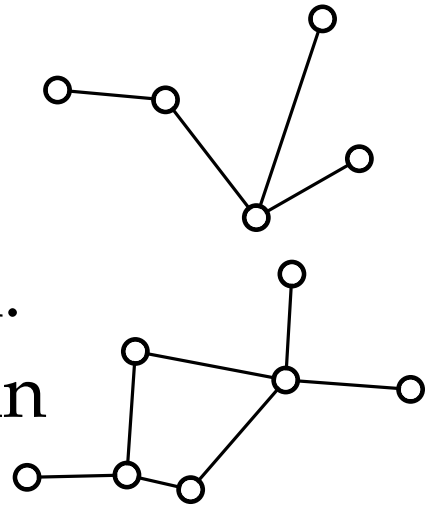
Lemma 3.
Der bipartite Graph $G = (\mathcal{A} \cup \mathcal{M}, E)$ ist ein Pseudo-Wald.

Extrempunkt-Lösungen haben $\leq |\mathcal{M}| + |\mathcal{A}|$ Variablen > 0 (L1).

Jede Komponente von G entspricht einer Extrempunkt-Lösung.

Lemma 4.
Der Graph H hat ein F-perfektes Matching.

Pseudo-Bäume and -Wälder



Pseudo-Baum: Zusammenhängender Graph mit Knotenmenge V wird und höchstens $|V|$ Kanten.

Ein Pseudo-Baum ist entweder ein Baum oder ein Baum mit einer zusätzlichlichen Kante.

Pseudo-Wald: Sammlung von disjunkten Pseudo-Bäumen

Lemma 3.
Der bipartite Graph $G = (\mathcal{A} \cup \mathcal{M}, E)$ ist ein Pseudo-Wald.

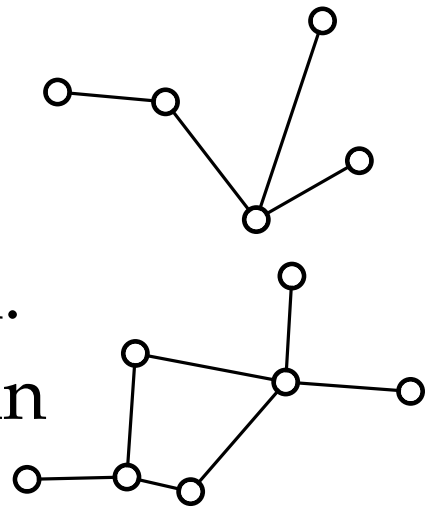
Extrempunkt-Lösungen haben $\leq |\mathcal{M}| + |\mathcal{A}|$ Variablen > 0 (L1).

Jede Komponente von G entspricht einer Extrempunkt-Lösung.

Lemma 4.
Der Graph H hat ein F-perfektes Matching.

H ist auch ein Pseudo-Wald: entferne 1 Kante pro $v \in \mathcal{A} \setminus F$

Pseudo-Bäume and -Wälder



Pseudo-Baum: Zusammenhängender Graph mit Knotenmenge V wird und höchstens $|V|$ Kanten.

Ein Pseudo-Baum ist entweder ein Baum oder ein Baum mit einer zusätzlichlichen Kante.

Pseudo-Wald: Sammlung von disjunkten Pseudo-Bäumen

Lemma 3.
Der bipartite Graph $G = (\mathcal{A} \cup \mathcal{M}, E)$ ist ein Pseudo-Wald.

Extrempunkt-Lösungen haben $\leq |\mathcal{M}| + |\mathcal{A}|$ Variablen > 0 (L1).

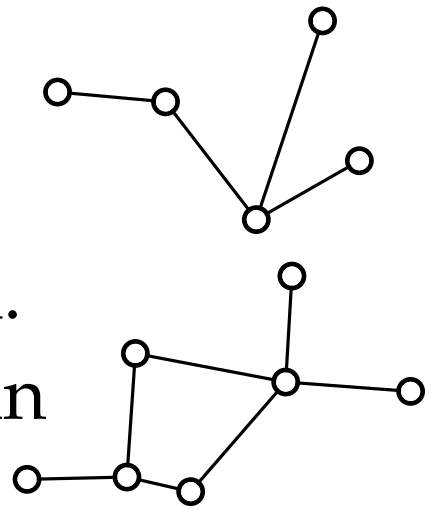
Jede Komponente von G entspricht einer Extrempunkt-Lösung.

Lemma 4.
Der Graph H hat ein F-perfektes Matching.

H ist auch ein Pseudo-Wald: entferne 1 Kante pro $v \in \mathcal{A} \setminus F$

Knoten in F haben min. Grad 2

Pseudo-Bäume and -Wälder



Pseudo-Baum: Zusammenhängender Graph mit Knotenmenge V wird und höchstens $|V|$ Kanten.

Ein Pseudo-Baum ist entweder ein Baum oder ein Baum mit einer zusätzlichlichen Kante.

Pseudo-Wald: Sammlung von disjunkten Pseudo-Bäumen

Lemma 3.
Der bipartite Graph $G = (\mathcal{A} \cup \mathcal{M}, E)$ ist ein Pseudo-Wald.

Extrempunkt-Lösungen haben $\leq |\mathcal{M}| + |\mathcal{A}|$ Variablen > 0 (L1).

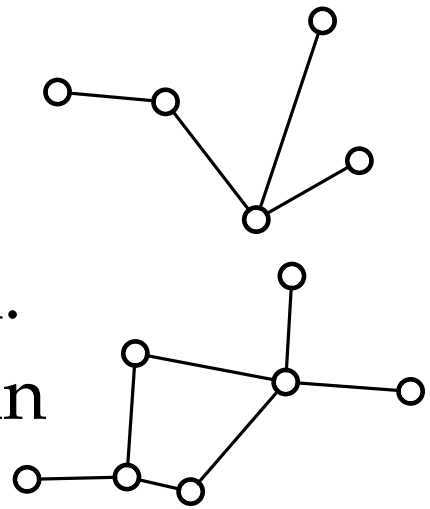
Jede Komponente von G entspricht einer Extrempunkt-Lösung.

Lemma 4.
Der Graph H hat ein F-perfektes Matching.

H ist auch ein Pseudo-Wald: entferne 1 Kante pro $v \in \mathcal{A} \setminus F$

Knoten in F haben min. Grad 2 $\Rightarrow$ Die Blätter in H sind Maschinen.

Pseudo-Bäume and -Wälder



Pseudo-Baum: Zusammenhängender Graph mit Knotenmenge V wird und höchstens $|V|$ Kanten.

Ein Pseudo-Baum ist entweder ein Baum oder ein Baum mit einer zusätzlichlichen Kante.

Pseudo-Wald: Sammlung von disjunkten Pseudo-Bäumen

Lemma 3.
Der bipartite Graph $G = (\mathcal{A} \cup \mathcal{M}, E)$ ist ein Pseudo-Wald.

Extrempunkt-Lösungen haben $\leq |\mathcal{M}| + |\mathcal{A}|$ Variablen > 0 (L1).

Jede Komponente von G entspricht einer Extrempunkt-Lösung.

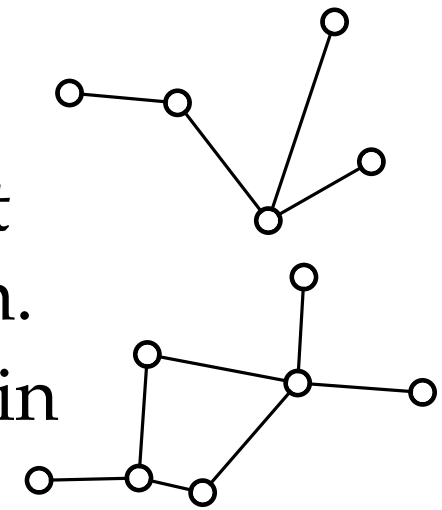
Lemma 4.
Der Graph H hat ein F-perfektes Matching.

H ist auch ein Pseudo-Wald: entferne 1 Kante pro $v \in \mathcal{A} \setminus F$

Knoten in F haben min. Grad 2 $\Rightarrow$ Die Blätter in H sind Maschinen.

Wenn man iterativ alle Blätter wählt,

Pseudo-Bäume and -Wälder



Pseudo-Baum: Zusammenhängender Graph mit Knotenmenge V wird und höchstens $|V|$ Kanten.

Ein Pseudo-Baum ist entweder ein Baum oder ein Baum mit einer zusätzlichlichen Kante.

Pseudo-Wald: Sammlung von disjunkten Pseudo-Bäumen

Lemma 3.
Der bipartite Graph $G = (\mathcal{A} \cup \mathcal{M}, E)$ ist ein Pseudo-Wald.

Extrempunkt-Lösungen haben $\leq |\mathcal{M}| + |\mathcal{A}|$ Variablen > 0 (L1).

Jede Komponente von G entspricht einer Extrempunkt-Lösung.

Lemma 4.
Der Graph H hat ein F-perfektes Matching.

H ist auch ein Pseudo-Wald: entferne 1 Kante pro $v \in \mathcal{A} \setminus F$

Knoten in F haben min. Grad 2 $\Rightarrow$ Die Blätter in H sind Maschinen.

Wenn man iterativ alle Blätter wählt, bleiben nur noch Kreise übrig.

Auftragsplanung auf parallelen Maschinen

Satz. Es gibt einen LP-basierten Faktor-2-Approximations-algorithmus für das Problem der Auftragsplanung auf unabhängigen parallelen Maschinen.

Auftragsplanung auf parallelen Maschinen

Satz. Es gibt einen LP-basierten Faktor-2-Approximations-algorithmus für das Problem der Auftragsplanung auf unabhängigen parallelen Maschinen.

Scharf?

Auftragsplanung auf parallelen Maschinen

Satz. Es gibt einen LP-basierten Faktor-2-Approximations-algorithmus für das Problem der Auftragsplanung auf unabhängigen parallelen Maschinen.

Scharf? Ja!

Instanz I_m:

Auftragsplanung auf parallelen Maschinen

Satz. Es gibt einen LP-basierten Faktor-2-Approximations-algorithmus für das Problem der Auftragsplanung auf unabhängigen parallelen Maschinen.

Scharf? Ja!

Instanz I_m:

m Maschinen und $m^2 - m + 1$ Aufträge

Auftragsplanung auf parallelen Maschinen

Satz. Es gibt einen LP-basierten Faktor-2-Approximations-algorithmus für das Problem der Auftragsplanung auf unabhängigen parallelen Maschinen.

Scharf? Ja!

Instanz I_m:

m Maschinen und $m^2 - m + 1$ Aufträge

Auftrag a_1 hat Bearbeitungszeit m auf allen Maschinen,

Auftragsplanung auf parallelen Maschinen

Satz. Es gibt einen LP-basierten Faktor-2-Approximations-algorithmus für das Problem der Auftragsplanung auf unabhängigen parallelen Maschinen.

Scharf? Ja!

Instanz I_m:

m Maschinen und $m^2 - m + 1$ Aufträge

Auftrag a_1 hat Bearbeitungszeit m auf allen Maschinen, alle anderen Aufträge haben Bearbeitungszeit 1 auf allen Maschinen.

Auftragsplanung auf parallelen Maschinen

Satz. Es gibt einen LP-basierten Faktor-2-Approximations-algorithmus für das Problem der Auftragsplanung auf unabhängigen parallelen Maschinen.

Scharf? Ja!

Instanz I_m:

m Maschinen und $m^2 - m + 1$ Aufträge

Auftrag a_1 hat Bearbeitungszeit m auf allen Maschinen, alle anderen Aufträge haben Bearbeitungszeit 1 auf allen Maschinen.

Optimum: eine Maschine mit a_1, Rest gleichmäßig verteilt.

Auftragsplanung auf parallelen Maschinen

Satz. Es gibt einen LP-basierten Faktor-2-Approximations-algorithmus für das Problem der Auftragsplanung auf unabhängigen parallelen Maschinen.

Scharf? Ja!

Instanz I_m:

m Maschinen und $m^2 - m + 1$ Aufträge

Auftrag a_1 hat Bearbeitungszeit m auf allen Maschinen, alle anderen Aufträge haben Bearbeitungszeit 1 auf allen Maschinen.

Optimum: eine Maschine mit a_1, Rest gleichmäßig verteilt.

Algorithmus:

Auftragsplanung auf parallelen Maschinen

Satz. Es gibt einen LP-basierten Faktor-2-Approximations-algorithmus für das Problem der Auftragsplanung auf unabhängigen parallelen Maschinen.

Scharf? Ja!

Instanz I_m:

m Maschinen und $m^2 - m + 1$ Aufträge

Auftrag a_1 hat Bearbeitungszeit m auf allen Maschinen, alle anderen Aufträge haben Bearbeitungszeit 1 auf allen Maschinen.

Optimum: eine Maschine mit a_1, Rest gleichmäßig verteilt.

Algorithmus:

LP(T) hat keine zulässige Lösung für $T < m$.

Auftragsplanung auf parallelen Maschinen

Satz. Es gibt einen LP-basierten Faktor-2-Approximations-algorithmus für das Problem der Auftragsplanung auf unabhängigen parallelen Maschinen.

Scharf? Ja!

Instanz I_m:

m Maschinen und $m^2 - m + 1$ Aufträge

Auftrag a_1 hat Bearbeitungszeit m auf allen Maschinen, alle anderen Aufträge haben Bearbeitungszeit 1 auf allen Maschinen.

Optimum: eine Maschine mit a_1, Rest gleichmäßig verteilt.

Algorithmus:

LP(T) hat keine zulässige Lösung für $T < m$.

Extrempunkt-Lösung: Weise $1/m$ von a_1 und $m - 1$ anderen Aufträge zu jeder Maschine zu.

Auftragsplanung auf parallelen Maschinen

Satz. Es gibt einen LP-basierten Faktor-2-Approximations-algorithmus für das Problem der Auftragsplanung auf unabhängigen parallelen Maschinen.

Scharf? Ja!

Instanz I_m:

m Maschinen und $m^2 - m + 1$ Aufträge

Auftrag a_1 hat Bearbeitungszeit m auf allen Maschinen, alle anderen Aufträge haben Bearbeitungszeit 1 auf allen Maschinen.

Optimum: eine Maschine mit a_1, Rest gleichmäßig verteilt.

Algorithmus:

LP(T) hat keine zulässige Lösung für $T < m$.

Extrempunkt-Lösung: Weise $1/m$ von a_1 und $m - 1$ anderen Aufträge zu jeder Maschine zu.

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

Auftragsplanung auf parallelen Maschinen

Satz. Es gibt einen LP-basierten Faktor-2-Approximations-algorithmus für das Problem der Auftragsplanung auf unabhängigen parallelen Maschinen.

Besser?

Auftragsplanung auf parallelen Maschinen

Satz. Es gibt einen LP-basierten Faktor-2-Approximations-algorithmus für das Problem der Auftragsplanung auf unabhängigen parallelen Maschinen.

Besser?

Es ist kein besserer Approximationsalgorithmus bekannt.

Auftragsplanung auf parallelen Maschinen

Satz. Es gibt einen LP-basierten Faktor-2-Approximations-algorithmus für das Problem der Auftragsplanung auf unabhängigen parallelen Maschinen.

Besser?

Es ist kein besserer Approximationsalgorithmus bekannt.

Das Problem lässt sich nicht mit Faktor $< 3/2$ approximieren (außer P=NP) [Lenstra, Shmoys & Tardos '90]

Auftragsplanung auf parallelen Maschinen

Satz. Es gibt einen LP-basierten Faktor-2-Approximations-algorithmus für das Problem der Auftragsplanung auf unabhängigen parallelen Maschinen.

Besser?

Es ist kein besserer Approximationsalgorithmus bekannt.

Das Problem lässt sich nicht mit Faktor $< 3/2$ approximieren (außer P=NP) [Lenstra, Shmoys & Tardos '90]

Bei konstanter Anzahl Maschinen gibt es für jedes $\varepsilon > 0$ einen Faktor-$(1 + \varepsilon)$-Approximationsalgorithmus.

[Horowitz & Sahni '76]

Auftragsplanung auf parallelen Maschinen

Satz. Es gibt einen LP-basierten Faktor-2-Approximations-algorithmus für das Problem der Auftragsplanung auf unabhängigen parallelen Maschinen.

Besser?

Es ist kein besserer Approximationsalgorithmus bekannt.

Das Problem lässt sich nicht mit Faktor $< 3/2$ approximieren (außer P=NP) [Lenstra, Shmoys & Tardos '90]

Bei konstanter Anzahl Maschinen gibt es für jedes $\varepsilon > 0$ einen Faktor-$(1+\varepsilon)$-Approximationsalgorithmus.
[Horowitz & Sahni '76]

Für uniforme Maschinen gibt es für jedes $\varepsilon > 0$ einen Faktor-$(1+\varepsilon)$-Approximationsalgorithmus.
[Hochbaum & Shmoys '87]

Auftragsplanung auf parallelen Maschinen

Satz. Es gibt einen LP-basierten Faktor-2-Approximations-algorithmus für das Problem der Auftragsplanung auf unabhängigen parallelen Maschinen.

Besser?

Es ist kein besserer Approximationsalgorithmus bekannt.

Das Problem lässt sich nicht mit Faktor $< 3/2$ approximieren (außer P=NP) [Lenstra, Shmoys & Tardos '90]

Bei konstanter Anzahl Maschinen gibt es für jedes $\varepsilon > 0$ einen Faktor-$(1+\varepsilon)$-Approximationsalgorithmus.
[Horowitz & Sahni '76]

Für uniforme Maschinen gibt es für jedes $\varepsilon > 0$ einen Faktor-$(1+\varepsilon)$-Approximationsalgorithmus.
(Maschinen unterschiedlich schnell, aber bearbeiten Aufträge uniform) [Hochbaum & Shmoys '87]