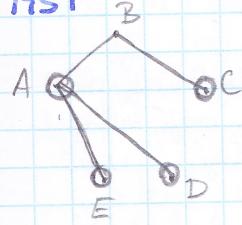


\vdash : MST



Převážení

A-D
C-E
cena je 4

C: ABCEDA

H: ABCEDA

čísla je deset

○ jsou všechny v W

Těžké úlohy: referativní cesty, loky, TSP. Nejčastěji lze úlohu formulace úloh jako cestu, loky, knapsack

Přednáška
23.4.2019

Rozvrhování - přiřazení úloh na stroje v čase, současně s plánováním

Typy rozvrhování

- na jednom stroji (jeden dělňák)
- na více stejných strojích (deset stejně pracujících dělňáků)
- na různých strojích (skupina kopací, sečníků, jízdníků atd.)

Nařízení:

- lákouna dce minimalizovat náklady na skladování barev, ale vše musí dodat včas. Navíc minimálně dočasné pracovní doby a respektovat výrobní procesy a pásťkové časy (číslované výrobní linky) atd.

Běžná formulace

- n úloh $T = \{T_1, T_2, \dots, T_n\}$ → kolabat daný stroj máme
- m strojů s kapacitami R_k , $P = \{P_1^1, \dots, P_{R_1}^1, P_1^2, \dots, P_{R_2}^2, \dots, P_1^m, \dots, P_{R_m}^m\}$
- chceme v čase úloham přiřadit stroje tak, aby všechny úlohy byly dokončeny, mimo byt zadány termín dokončení
- offline: dostaneme T a P a vše předem naplánijeme \leftarrow to se dělá v tomto přednášce
- online: získáme T , ale úlohy dorážejí a novějšíme se během \leftarrow nejdříve dělat
- výsledek se prezentuje Ganttovým diagramem - obdélníky

Termíni:

- obecně:
 - úloha může v jednom okamžiku spracovávat pouze jeden stroj
 - jeden stroj může pracovat růžy jen na jedné úloze
- specifické:
 - úlohy musí být správné v přidruženém intervalu $\langle r_i, d_i \rangle$ $\begin{matrix} \nearrow \text{release} \\ \searrow \text{deadline} \end{matrix}$
 - úlohy na sobě mohou být seřazeny; tj. můžou se udržet ve správném pořadí, třeba T_j začíná na T_i : $T_i < T_j$

r_i : release time

p_i : processing time

s_i : start time

c_i : completion time

d_i : due date, překročit je mít penalizováno

\bar{d}_i : deadline, nelze překročit nikdy

Grahamova notace ($\alpha / \beta / \gamma$) je)

zdroje | úlohy | kritéria

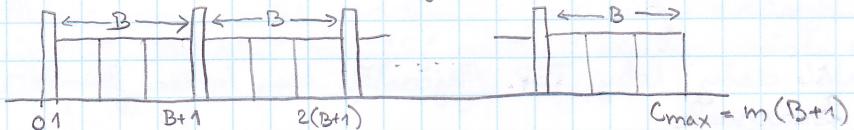
Typy strojů

- paralelní: všechny stroje můžou
- dedikované stroje: každý stroj dělá jinou nějakou činnost

Věta: Uloha $1/r_j, d_j | C_{max}$ je sice NP říká.

Dоказ: Uděláme redukci 3-partition problemu: Máme nádoby do každého vložit 3 předměty. Všechny nádoby mají velikost B , a předměty mají velikost a , $B/n < a_i < B/2$. O 3-partition problemu se ví, že je sice NP říká.

Zavedu užitku rozvrhování s jedním procesorem. Pro každou nádobu vytvořím



časový úsek délky B , a tyto úseky (nádoby) oddělujem moje výrobenými užitkami velikosti 1, které mohou být zahnuté jenom v jednu stranu.

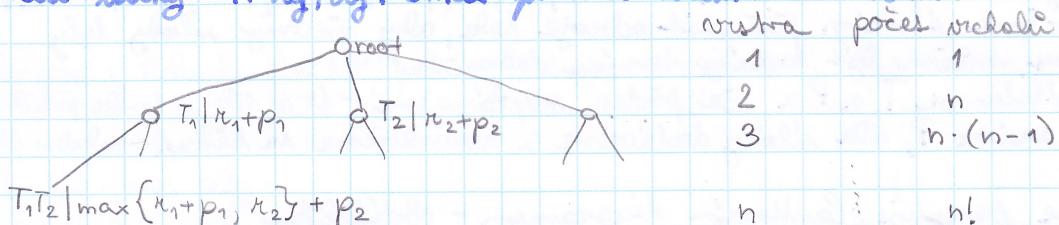
Pak pokud se podaří rozvrhnout užitky, máli jsme i 3 partition problem.

Formulace $1/r_j, d_j | C_{max}$ jako LP $\text{sg: slouží užitky } q$

$$\begin{array}{ll} \min C_{max} & \\ \text{s.t.} & \sum_{i=1}^n x_{iq} = 1, \quad \forall i = 1 \dots n \\ & \sum_{i=1}^n x_{iq} = 1, \quad \forall q = 1 \dots n \\ & t_{q-1} \geq \sum_{i=1}^n m_i \cdot x_{iq}, \quad \forall q = 1 \dots n \\ & t_q \geq t_{q-1} + \sum_{i=1}^n p_i \cdot x_{iq}, \quad \forall q = 2 \dots n \\ & t_q \leq \sum_{i=1}^n d_i \cdot x_{iq} - \sum_{i=1}^n p_i \cdot x_{iq}, \quad \forall q = 1 \dots n \\ & C_{max} \geq t_n + \sum_{i=1}^n p_i \cdot x_{in} \end{array}$$

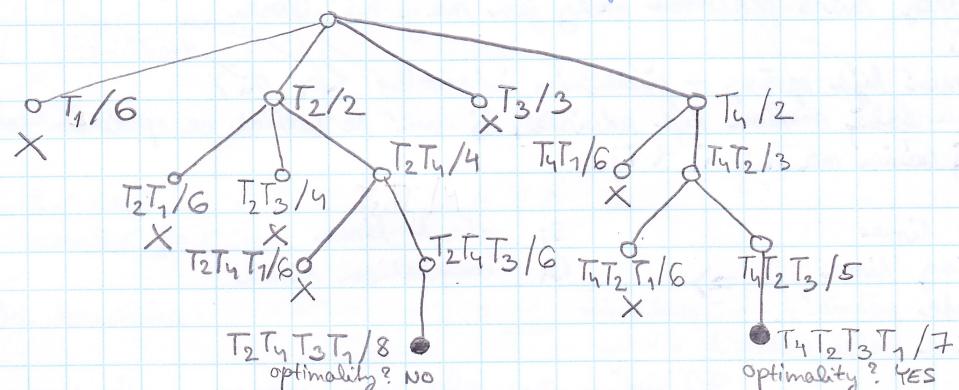
užitka je jen na jeho pozici
na každé pozici je jen jedna užitka
začneme až po překonávání
užitka je jen na jeho pozici
užitka je jen na jeho pozici
začneme až po překonávání

Rешení užitky $1/r_j, d_j | C_{max}$ pomocí branch and bound



-bounding: se nezávádí, když vědíme, že se nám tam slíží minimálně výška

Příklad: Bradleyho algoritmus: $r = (4, 1, 1, 0)$, $p = (2, 1, 2, 2)$, $d = (8, 5, 6, 4)$



Na konci stranu testujeme optimalitu - více slide 20/83 v shed-e.pdf

Užitka, kde minimalizujeme $\sum w_j \cdot C_j$ - vyplňte se nejdříve výše užitky nejrychlejší užitky na zadánku (SPT = shortest path first)

Formulace 1 | prec | ZwiCi jako LP:

$$x_{ij} = 1 \text{ iff } i=j \text{ nebo } T_j \text{ je předchůdcem } T_i, \text{ jinak } x_{ij} = 0 \quad \leftarrow \text{proměnná}$$

$$e_{ij} = 1 \text{ iff } i=j \text{ nebo v grafu s kružnicí je hrana z } T_i \text{ do } T_j, \text{ jinak } e_{ij} = 0 \quad \leftarrow \text{sídlový graf}$$

$$\min \sum_{i=1}^n \sum_{j=1}^n p_i \cdot w_j \cdot x_{ij}$$

$$\text{st: } x_{ij} \geq e_{ij}, \quad i, j \in \{1, \dots, n\}$$

$$x_{ij} + x_{ji} = 1, \quad i, j \in \{1, \dots, n\}, \quad i \neq j \quad \text{buk } j \text{ je } i \text{ před } j, \text{ nebo } j \text{ před } i, \text{ ne oba}$$

$$1 \leq x_{ij} + x_{jk} + x_{ki} \leq 2, \quad i, j, k \in \{1, \dots, n\}, \quad i \neq j \neq k \quad \text{neexistuje tam cyklus}$$

$$x_{ii} = 1, \quad i \in \{1, \dots, n\}$$

TSP: na vstupe je úplný graf $G = (V, E)$ a cílový hran $c: E \rightarrow \mathbb{R}^+$
výsledek je kružnice s minimálním časem, kružnice vede přes všechny vrcholy.

Cvičení
25.4.2019

Obecné TSP je NPúplné, a neexistuje na něj apoximační algoritmus. Na upravené verze TSP nějaké algoritmy existují - běba-TSP splňující Δ nerovnost.

Model TSP: $x_{ij} \in \{0, 1\}, \quad x_{ij} = 1 \text{ iff } \text{hrana } i, j \text{ leží na trase cestujícího}$

$$\sum_{j \in V} x_{ij} = 1 \quad \forall i : \text{do vrcholu } j \text{ vstoupí právě jeden}$$

$$\sum_{i \in V} x_{ij} = 1 \quad \forall j : \text{z vrcholu } i \text{ vystoupí právě jeden}$$

S_i : čas vstupu $S_i + c_{ij} \leq S_j + M(1 - x_{ij})$ Do vrcholu i vstoupí v čase S_i , do vrcholu j a čas S_j nastane později (c_{ij} jsou kladné). Nemůže mít platit pro hrany, které jsem použila ($i, j : x_{ij} = 1$). Podmínka nemůže platit pro první vrchol (jinak by cyklus neléhal učinit)

$$\min \sum c_{ij} \cdot x_{ij}$$

Jiný model

$$\min \sum c_{ij} \cdot x_{ij}$$

použijeme představu, že obchodník má n výrobků a v každém městě jeden nechá. Zavedeme pomocnou proměnnou $y_{ij} \in \mathbb{N}$, která vyjadřuje, kolik sběr má obchodník na trase i do j

$$\sum_{j \in V} x_{ij} = 1 \quad \forall i$$

nezpravidelné hrany $\Rightarrow y_{ij} \leq (n-1) \cdot x_{ij}$

nezpravidelné sběry $\Rightarrow y_{ij} \leq (n-1) \cdot x_{ij}$

ještě se jedná $\Rightarrow (\sum_{(i,j) \in E} y_{ij}) - 1 = \sum_{(j,k) \in E} y_{jk} \quad \forall j \in V - \{i\}$

obči můžete $\sum_{j \in V} y_{ij} + (n-1) = \sum_{j \in V} y_{jk}$ na začátku máme $n-1$ sběrů, na konci 0.

První model spočítal TSP na 50 vrcholech s gap 1.77% a na 70 vrcholech s gap 18.39%. Druhý model na 50 i 70 vrcholech najde optimální řešení a na 100 vrcholech má gap 4.43%. (po 30s běhu)

Ještě jiný způsob vyhledání kružnice: $\sum_{(i,j) \in E} x_{ij} \leq |S| - 1$ pro každou kružnici S , $|S| < n$

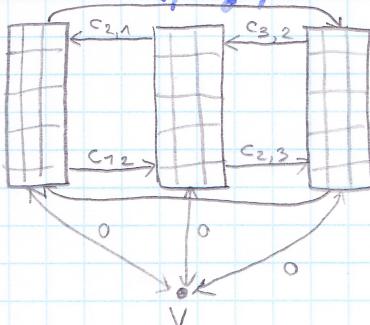
- jenž je to máme kružně moc omezení

- řeší se to tak, že najdeme nějaké řešení, a nám kontrolujeme, jestli tam jsou male kružnice (to je v p čase pomocí DFS/BFS)

- pokud nějakou kružnici najdeme, přímo ji zakážeme, takže nemůžeme vytvořit generovaná všechna omezení

- tato metoda vyřešila TSP na 50, 70, 100 i 159 vrcholech. Na 260 vrcholech mi to trvalo přes minutu

Domácí úkol 4: máme proužky pixelů z obrázku. Chceme obrázek rozšiřit tak, aby pixelové hodnoty sousedních pruhů byly minimální



Zavedeme nový vrchol V , který zajistí, abychom nepermalizovali rozdíl mezi levým a pravým okrajem obrázku.

Pozor, $c_{ij} \neq c_{ji}$, protože proužky mají šířku mezi pixelů.