

- 3) Najdeme pohováru s nízkou cenou. Dělá se to tak, že v matici kreslíme všechny a nízké ceny a smažeme se zkrátit všechny muly. Pokud to bude udělat méně než n čarami, tak pohováru neexistuje, jinak pustíme bod (4).
- 4) Ve sloupech se hledají následující minima A, B, ale snad je: Najdeme minimum z nekruhových hodnot, označme ho m. Od všech nekruhových čísel odečteme m. Když všechna původní čísla čaram přičteme m, zdejší řádky čísla se rovnají. Opakujeme krok (3).
- 5) V grafu sedí je pohováru s nízkou cenou. Tykem stejně hrany v původním grafu, speciálně jejich cenu, a máme, že levnější pohováru tam najdeme.

Dobře je to popsané ve videu Hungarian Method od Kaiser Wise na youtube.

Knapsack problem: Máme n předmětů, máme jejich ceny  $c_1, \dots, c_n$ , a jejich váhy  $w_1, \dots, w_n$ . Balík má maximální hmotnost  $W$ , kterou nechceme překročit. Chceme dát balohnu daň - ty předměty, aby byla cena předmětů maximální.

Fractional knapsack: Relativně (vykáslena se na detaily) užíváme balohnu, když nejsou všechny předměty. Postavíme se z LP do IP:

$$\max \sum_{i=1}^n x_i \cdot c_i$$

$$\text{s.t. } \sum_{i=1}^n x_i \cdot w_i \leq W$$

$$x \in \{0, 1\}$$

Jak se to daří řešit? Vypočítáme "relativní ceny"  $\frac{c_i}{w_i}$ . Tj. máme cenu na jednotku váhy. Předměty seřadíme podle relativní ceny a balohnu užíváme nejdrahšími předměty. Jakmile dojde k předmětu, který se tam nevuje celý, rozdělime ho a do balohnu vložíme jen část, kolik je platná jeho cena - nám to nejdřív včetně.  $O(n \cdot \log(n))$

proximaci: alg. pro řešení úloh bylo složitě.

$$n \cdot J^*(I) \geq J^k(I) \geq \frac{1}{n} \cdot J(I)$$

$J^*$ : optimální řešení

$J^k$ : řešení s heuristikou A

Algoritmus A je  $n$ -approximace, když vzdálenost platí pro  $n \geq 1$

Knapsack lze řešit dynamickým programováním (pseudopolynomial:  $O(n \cdot C)$ )

Přednáška  
16.4.2019

$$\exists: w = (21, 35, 52, 17), c = (10, 20, 30, 10), W = 100$$

- správně když už delší sloupec pro každé číslo, ale tady vidíme, že už jde jednoduše na desetitiskálovce, protože jiné hodnoty a tam neobjeví

	0	10	20	30	40	50	60	70	80	90	...	$\sum c_i$
0	0	$\infty$	...	při expozici bunky v řádku								
1	0	$\infty$	i-1 máme dvě varianty:									
2	0	21	$\infty$	1) přidáme do bunky níže hodnotu, která i jsem tam nedala								
3	0	21	35	$\infty$	2) prvek i tam zůstane, přičtené k hodnotě už i a jsem sloupec o $c_i$							
4	0	17	35	52	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$		

- poté můžeme létat v řádku dvě čísla, vybereme to menší

jmenem sloupec o  $c_i$

- po konci algoritmu je řešení 10 čísla, které je nejvíce vpravo

+

- hodnoty, které překážejí  $W$ , vše dále neexpandujeme

strukturně to je dělat s provozem  $c$  a  $w$ , ale v bunkách budeme maximizovat

	0	1	2	3	4	5	6	7	8	$= W$
0	0	$\infty$								
1	0	$\infty$	3.1	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$	
2	0	$\infty$	3.1	4.2	$\infty$	7.3	$\infty$	$\infty$	$\infty$	
3	0	$\infty$	3.1	4.2	5.1	7.3	8.2	9.3	$\infty$	$\rightarrow \text{out}, W=9.3$
4	0	$\infty$	3.1	4.2	5.1	7.3	8.2	9.3	8.5	

Prí: Snažíme se vordlit lyže na kousky, a chceme co nejméně odpaď

- odděluj řešení níkol, ale řešíme dely

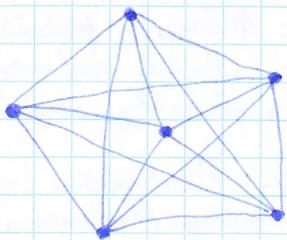
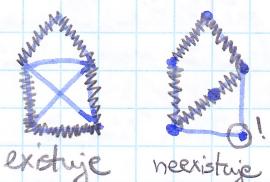
- máme krabice různých rozměrů a chceme je skládat na sebe, mítéme nějaká omezení: např: ře neležit krabice na krabici

### Travelling salesman problem - TSP

- na obecnou úlohu nelze najít ani apotimaciální algoritmus

Máme mapu měst, mezi nimi jsou cesty s různými cenami, graf je splný. Obchodní cestující navštíví všechna města a máte ho na start, aby byla cesta co nejlevnější

### Existence Hamiltonovské kružnice



TSP hledá v splném grafu kom. kružnici tak, aby byla minimální (nebo méně než násobek K)

Def: L je úloha, P je polynom, L<sub>P</sub> je úloha omezená jen na instance L, kde  $\text{larger}(L) \leq p(\text{size}(L))$ .  
Ukážeme, že úloha L je silně NP řešitelná, když L<sub>P</sub> je NP řešitelná pro nějaký polynom p.

Ukázání: TSP je silně NP řešitelná.

Dokazuje se to tak, že omezíme ceny hran na čísla 1 a 2. Polud lze takto ak nyní řešit, nyní řešit Hamiltonovskou kružnicí, a to je NP. Proto musí být i TSP s hranami 1,2 NP řešitelný.

Ukázání: Pokud  $P \neq NP$ , pak neexistuje apotimaciální algoritmus.

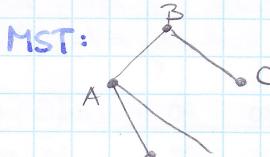
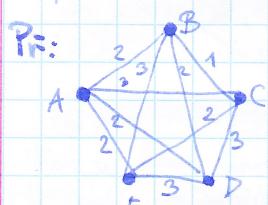
Naopak ho je stejně řešitelné jako řešit NP řešitelnou úlohu, díkyž je v TALECH.

### Double-tree alg.:

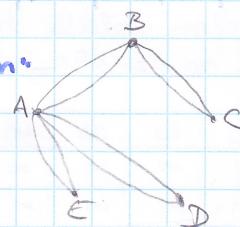
- najde minimální košť

- zdvojíme hranu v košti, najdeme Eulerovský tah

- převédeme Eulerovský tah na Hamiltonovskou kružnici



zdvojené hrany:



Eulerovská cesta: A B C B A D A E A

Hamiltonovská kružnice: A B C D E A

Hodnota košťnice:  $2+1+3+3+2 = \underline{\underline{11}}$

Double-tree alg. je apotimaciální alg. pro metriku TSP  
precněji, je to 2-apotimaciální algoritmus

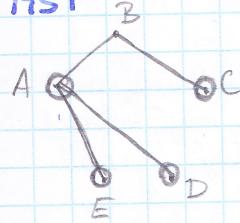
### Christofides alg.

- najde minimální košť

- najde minimum W vzhledu lichého stupně v košti (kde jich má říká počet)

- najde parsování M vzhledu ve W

$\vdash$ : MST



Převážení

A-D  
C-E  
cena je 4

C: ABCEDA

H: ABCEDA

cena je deset

○ jsou všechny v W

T řešení bude: referativní cesty, loky, TSP. Nejčastěji lze řešit formulací siloh jako cest, loky, knapsack

Rozvrhování - přiřazení siloh 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 kopáčů, rodiček, jížníků atd.)

Nařízení:

- lákona dle minimalizace náklady na skladování barev, ale vše musí dodat včas. Navíc minimálně dodržet pracovní dobu a respektovat výrobní procesy a pásťkové časy (číslované výrobní linky) atd.

Běžná formulace

- n siloh  $T = \{T_1, T_2, \dots, T_n\}$  → kolabovat daný stroj máme
- m strojů s kapacitami  $R_k$ ,  $P = \{P_1^1, P_1^2, \dots, P_{R_1}^1, P_1^2, \dots, P_{R_1}^m, P_2^1, \dots, P_{R_m}^1, P_2^2, \dots, P_{R_m}^m\}$
- chceme v čase silohem přidělit stroje tak, aby všechny silohy byly dokončeny, mimož výstupy během termínu dokončení
- offline: dostaneme  $T$  a  $P$  a vše předem naplánijeme  $\leftarrow$  to se dělá v kontrole přednášky
- online: enšíme  $P$ , ale silohy dorážejí a nově hýme za běhu.  $\leftarrow$  někde dělat
- výsledek se prezentuje Ganttovým diagramem - obdélníky

Intenzivní:

- obecná:
  - siloh může v jednom okamžiku spracovávat pouze jeden stroj
  - jeden stroj může pracovat různy jen na jedné silohě
- specifické:
  - silohy musí být spuštěny v přiděleném intervalu  $\langle r_i, d_i \rangle$  → release deadline
  - silohy na sobě mohou být rekvírovány, tj. mohou se nadelelat ve stejném pořadí, třeba  $t_j$  nadelel 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 | silohy | kritéria

Typy strojů

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

Přednáška

23.4.2019