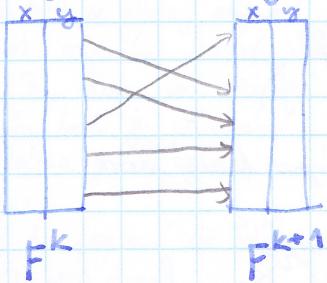


## Krok: Object tracking

Na vstupu je mnoha frames:  $\{F^{(1)}, F^{(2)}, \dots, F^{(p)}\}$  pro časy  $1 \dots p$ ,  
a  $F^k$  je tabulka  $x,y$  souřadnic.

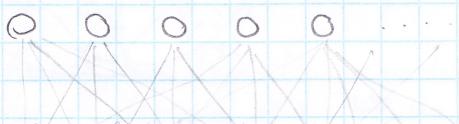


Hledáme hřejčí mení snímky,  
když kde se hře pohybuje

$$\text{Hledejme min } \sum_{i=1}^n d_{ij}$$

Budeme ho řešit jeho min cost pro dvoují sousedních snímků.

Snímek  $k$ :



vrchol pro každého  
hráče,  $b=1$

Snímek  $k+1$ :



vrchol pro každého  
hráče,  $b=-1$

Hrácy mají omezení  $\langle 0, 1 \rangle$ , a cena je vzdálenost bodů.

Můžeme ho dělat i Martinským algoritmem, ale implementace tady toho  
bude mnohem rychleji.

Min-cost flow: máme vstup  $(G, l, u, c, b)$ ,  $l, u: E(G) \rightarrow \mathbb{R}_+$ ,  $c: E(G) \rightarrow \mathbb{R}$ ,  
 $b: V(G) \rightarrow \mathbb{R}$ ,  $\sum_v b(v) = 0 \quad \forall v \in V$ .

Přednáška  
9.4.2019

Hledejme najít  $f$ , aby:  $\min \sum_{e \in E(G)} f(e) \cdot c(e)$

Formulace stejný jako LP:  $\min \sum_{e \in E(G)} f(e) \cdot c(e)$

$$\text{s.t. } \begin{aligned} \sum_{e \in S(v)} f(e) - \sum_{e \in S^-(v)} f(e) &= b(v) \quad \forall v \in V \\ l(e) &\leq f(e) \quad \forall e \in G(E) \\ f(e) &\leq u(e) \quad \forall e \in G(E) \end{aligned}$$

Max flow  $\Leftrightarrow$  min cost flow (viz předešlá strana)

Pro: Chinese postman problem: máme město s jednosměrnými ulicemi. Potřebuje projít všechny ulice a vrátit se na počátek.

$$b(v) = 0 \quad \forall v \in V$$

$$l(e) = 1, u(e) = \infty \quad \forall e \in E$$

Pokud lze horní čísla Eulerovskou cestou, dáme  $u(e) = 1 \quad \forall e \in E$

## Cycle cancelling algorithm

- 1) Najde nějaký přípustný kol.
- 2) Vytvoří se residuální graf (viz další strana)
- 3) V residuálním grafu se najde cyklus se zápornou cenou (cyklus omezující C)
- 4) Vypočítáme  $g_e = \min_{e \in E(C)} u_f(e)$ , tj. o kolik lze nadefinovat kol, než můžeme  
na omezení vzhledově srovnat
- 5) Zvýšíme hodnotu v brancích na C podle toho, jak vyslo g.

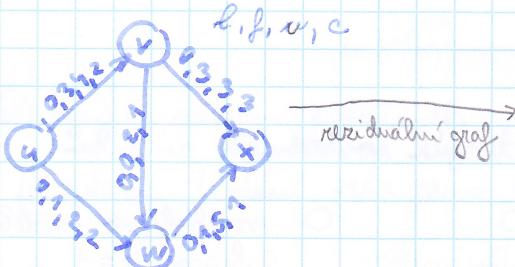
% funguje to jako Ford Fulkerson, ale místo slepsujících cest to hledá slepsující  
červené v residuálním grafu.

Zákon je  $O(|V| \cdot |E|^2 \cdot \max c_i \cdot \max u_i)$ , jež záleží na  $O(|V|^2 \cdot |E|^3 \cdot \log |V|)$

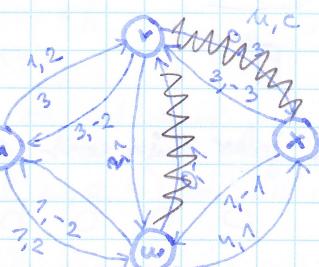
Residualní graf: Moží se ze všeho a abstraktního řešení. Ukažuje, jak lze ještě řešit vylopatit.

- vrcholy jsou na tom stejném místě v sítě
- nemáme hranu a dále je digraf, očtemené čísla směry
  - hranu vpravo mají horní směrem:  $u(i,j) = u(i,j) - f(i,j)$
  - hranu proti směru —————:  $u(i,j) = f(i,j) - l(i,j)$
- odstraníme hranu s nulovým horním směrem
- ceny hran nastavíme jako c pro hranu vpravo a -c pro hranu vlevo

Př:



residualní graf



hrany s  $u=0$   
obrácené

### Minimum cost multicommodity flow

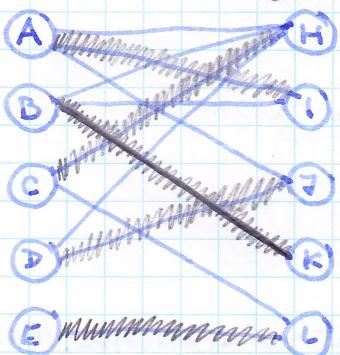
- máme několik různých komodit, každá má vlastní řadu  $f_m$

Např. po železnici vozíme různé sbory, ale každý má kapacitu v počtu vagónů (vleček) vstupem je  $(G, l, u, c, b^1, \dots, b^{M_1})$ , kde  $b^j$  jsou limitace komodity  $j$

$$\begin{aligned} \min & \sum_{e \in E(G)} \sum_{m \in M} f^m(e) \cdot c(e) \\ \text{s.t.} & \sum_{e \in S^+(v)} f^m(e) - \sum_{e \in S^-(v)} f^m(e) = b^m(v), \quad v \in V(G), m \in M \\ & l(e) \leq \sum_{m \in M} f^m(e) \leq u(e), \quad e \in E(G) \end{aligned}$$

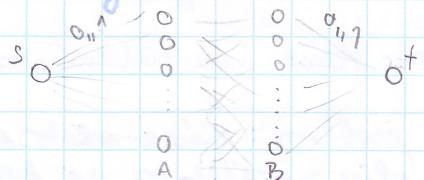
Úloha párování: Hledáme podmnožinu hran v grafu, aby každý vrchol měl pouze jeden hranu v gráfu

Pokud mají všechny vrcholy měřítkem hran v párování, je to perfektní párování



př: kdo dle slyším v lantech, když ne všechny páry jsou všechny společně lanty?

V bipartitním grafu můžeme problem formulovat jako maxflow



Hledání největšího párování: M-alternativní cesta

1) najdeme libovolné párování M

2) najdeme M-alternativní cestu, tj.  $E(P) - M$  je párování. Cesta je M-slepějící, protože koncové vrcholy P nejsou pokryty v M.

pridáme krajiny  
hrany

Jehož vše nesouží M-slepějící cesta, máme maximální párování!

Madarský algoritmus - nebrádí u šoubůsky

verze: graf G s dvěma množinami vrcholů X, Y,  $X \cap Y = \emptyset$ ,  $X \cup Y = V$ ,  $|X| = |Y| = n$ , a matice cen hran  $c_{ij}$ :  $c_{ij}$

postup: 1) Pro každý vrchol  $v \in X$  se spočítá  $p_i^* = \min_{j \in Y} \{c_{ij}\}$ , a pro každý  $v \in Y$  se spočítá  $p_j^* = \min_{i \in X} \{c_{ij} - p_i^*\}$

2) spočítá se graf GP:  $E = \{(i,j) \in E(G); c_{ij} - p_i^* - p_j^* = 0\}$

→ získáme malici, kde jsou měřítky (v každém řádku a sloupci aponí jedna)

- 3) Najdeme párování s nulovou cenou. Dělá se to tak, že v matici kreslíme svíle a nulové čísla a smažeme se stříhnout všechny svíly. Pokud to lze udělat méně než n čísly, tak párování neexistuje, jinak působíme bod (4).
- 4) Te slídech se hledají nějaké mnoiny A, B, ale snad je: Najdeme minimum z neštítných hodnot, označme ho m. Od všech neštítných čísel odečteme m. Ké všem číslům na prvním vrchu čen přičteme m, zdejší řádku čísla se rovnou. Opakujeme krok (3).
- 5) V grafu sedí je párování s nulovou cenou. Tykem stejně hrany v původním grafu, speciálne jejich cenu, a máme, že levnější párování tam najdeme.

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

Knapsack problem: Máme n předmětů, máme jejich ceny  $c_1, \dots, c_n$ , a jejich váhy  $w_1, \dots, w_n$ . Bohužel máme maximální hmotnost  $W$ , kterou nechceme překročit. Chceme dát bohatství předmětů, aby byla cena předmětů maximální.

Fractional knapsack: Relativně (vykáslena a na detaily) užívám metodu, když nejdou dělit předměty. Postavíme se z L.P. do L.P.:

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

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

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

Jak se to dát řešit? Vypočítáme "relativní ceny"  $\frac{c_i}{w_i}$ . Tj. máme cenu na jednotku váhy. Předměty seřadíme podle relativní ceny a pak si je vypíšeme. Nejdřív nejvýhodnějšími předměty. Jakmile dojde k předmětu, který se kam nevejdé celý, rozdělime ho a do bohatství vložíme jin část, když je platí a jsou v něm by výhodnější věci.  $O(n \cdot \log(n))$

proximační algoritmus pro řešení úlohy bohatství.

$$r \cdot J^*(I) = J^*(I) \geq \frac{1}{r} \cdot J(I)$$

$J^*$ : optimální řešení.

$J^A$ : řešení s heuristikou A

Algoritmus A je r-approximační, když vzdálost výsledku plati pro  $r \geq 1$

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

Přednáška  
16.4.2019

$$\text{Z: } w = (21, 35, 52, 17), c = (10, 20, 30, 10), W = 100$$

- správně když už delší sloupec pro každé číslo, ale tady vidíme, že už všichni jde záhadně na desátého sloupcem, protože jiné hodnoty a tam neobjeví

	0	10	20	30	40	50	60	70	80	90	...	$\sum c_i$
0	0	$\infty$	...	$C = \sum_{i=1}^n c_i$								
1	0	$\infty$	...	při expozici linií v řádku								
2	0	21	$\infty$	...	i-1 máme dvě varianty:							
3	0	21	35	$\infty$	...	1) přidáme do linií nás vše hodnotu, která i jsem tam nedali						
4	0	21	35	52	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$	...	2) přidáme i tam akce měl, přičtené k hodnotě $w_i$ a
5	0	17	35	52	69	$\infty$	$\infty$	$\infty$	$\infty$	$\infty$	...	přidáme sloupec o $c_i$

- poté můžeme lze i v řádku dvě čísla, vybereme to menší

poněme sloupec o  $c_i$

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

...

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

obdobně to je dělat s prvotním  $c$  a  $w$ , ale v řádkách 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	$\infty$	...
4	0	$\infty$	3.1	4.2	5.1	7.3	8.2	9.3	8.5	out, $W = 9.1$