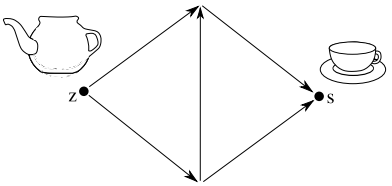
2. Toky v sítích

Představme si, že by v budově fakulty na Malé Straně existoval čajovod, který by rozváděl čaj do všech učeben. Znázorněme si to orientovaným grafem, v němž jeden významný vrchol představuje čajovar a druhý učebnu, ve které sedíme. Hrany mezi vrcholy pak znázorňují větvící se trubky, které mají čaj rozvádět. Chceme dopravit co nejvíce čaje do dané učebny, ale není to tak snadné, protože každá trubka má omezenou kapacitu – za jednotku času jí může protéci jen určité množství čaje, pro každou trubku obecně jiné. Může se tedy vyplatit za „tlustou" trubkou tok čaje rozvětvit a dál pokračovat více tenkými trubkami atd.



Čajovod

K podobnému problému dojdeme, budeme-li studovat přenos dat v počítačových sítích. Roli trubek zde hrají přenosové linky, kapacita říká, kolik dat přenesou za sekundu. Linky jsou spojené pomocí routerů a opět chceme dopravit co nejvíce dat z jednoho místa v síti na druhé. Data sice na rozdíl od čaje nejsou spojitá (přenášíme je po bytech nebo rovnou po paketech), ale při dnešních rychlostech přenosu je za taková můžeme považovat.

Toky v sítích

Definice: *Síť* je uspořádaná pětice (V, E, z, s, c), přičemž:

- (V, E) je orientovaný graf.
- $c : E \to \mathbb{R}_0^+$ je funkce přiřazující hranám jejich *kapacity*.
- $z, s \in V$ jsou dva vrcholy grafu, kterým říkáme *zdroj* a *stok* (neboli *spotřebič*).
- Graf je symetrický – je-li uv hranou grafu, je jí i vu.[⟨1⟩] Kdyby některá z opačných hran chyběla, můžeme ji přidat s nulovou kapacitou.

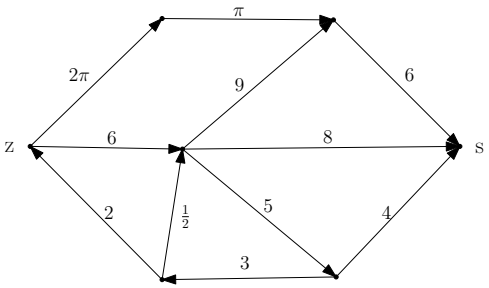
Definice: *Tok* je funkce $f : E \to \mathbb{R}_0^+$, pro níž platí:

1. Tok po každé hraně je omezen její kapacitou: $\forall e \in E : f(e) \leq c(e)$.

[⟨1⟩] Nebude-li hrozit nedorozumění, budeme hranu z vrcholu u do vrcholu v značit uv namísto formálnějšího, ale méně přehledného (u, v). Podobně u neorientovaných hran píšeme uv namísto $\{u, v\}$.

2. *Kirchhoffův zákon:* Do každého vrcholu přiteče stejně, jako z něj odteče („síť těsní“). Jedinou výjimku tvoří zdroj a spotřebič. Formálně:

$$\forall v \in V \setminus \{z, s\} : \sum_{u:uv \in E} f(uv) = \sum_{u:vu \in E} f(vu).$$



Příklad sítě. Čísla představují kapacity jednotlivých hran.

Sumy podobné těm v Kirchhoffově zákoně budeme psát často, zavedeme si pro ně tedy šikovné značení:

Definice: Pro libovolnou funkci $f : E \to \mathbb{R}$ definujeme:

- $f^+(v) := \sum_{u:uv \in E} f(uv)$ (celkový *přítok* do vrcholu)
- $f^-(v) := \sum_{u:vu \in E} f(vu)$ (celkový *odtok* z vrcholu)
- $f^\Delta(v) := f^+(v) - f^-(v)$ (*přebytek* ve vrcholu)

(Kirchhoffův zákon pak říká prostě to, že $f^\Delta(v) = 0$ pro všechna $v \neq z, s$.)

Pozorování: V každé síti nějaký tok existuje: třeba funkce, která je všude nulová (po žádné hraně nic neteče). To je korektní tok, ale sotva užitečný. Budeme chtít najít tok, který přepraví co nejvíce tekutiny ze zdroje do spotřebiče.

Definice: *Velikost toku* f budeme značit $|f|$ a položíme ji rovnu součtu velikostí toku na hranách vedoucích do spotřebiče minus součet velikostí toků na hranách ze spotřebiče ven. V naší terminologii je to tedy přebytek ve spotřebiči: $|f| := f^\Delta(s)$.

Pozorování: Jelikož síť těsní, mělo by být jedno, zda velikost toku měříme u spotřebiče nebo u zdroje. Vskutku, krátkým výpočtem ověříme, že tomu tak je:

$$f^\Delta(z) + f^\Delta(s) = \sum_v f^\Delta(v) = 0.$$

První rovnost platí proto, že podle Kirchhoffova zákona jsou zdroj a spotřebič jediné dva vrcholy, jejichž přebytek může být nenulový. Druhou rovnost získáme tak, že si uvědomíme, že tok na každé hraně přispěje do celkové sumy jednou s kladným znaménkem a jednou se záporným. Zjistili jsme tedy, že přebytek zdroje a spotřebiče se liší pouze znaménkem. ♡

Poznámka: Rádi bychom nalezli v zadané síti tok, jehož velikost je maximální. Máme ale zaručeno, že maximum bude existovat? Všech možných toků je nekonečně mnoho a v nekonečné množině se může snadno stát, že ačkoliv existuje supremum, není maximem (příklad: $\{1 - 1/n \mid n \in \mathbb{N}^+\}$). Odpověď nám poskytne matematická analýza: množina všech toků je kompaktní podmnožinou prostoru $\mathbb{R}^{|E|}$, velikost toku je spojitá (dokonce lineární) funkce z této množiny do $\mathbb{R}$, takže musí nabývat minima i maxima.

Nám ale bude stačit studovat sítě s racionálními kapacitami, kde existence maximálního toku bude zjevná už z toho, ze sestrojíme algoritmus, který takový tok najde.

Fordův-Fulkersonův algoritmus

Nejjednodušší z algoritmů na hledání maximálního toku pochází od Forda a Fulkersona. Je založen na prosté myšlence: začname s nulovým tokem a postupně ho budeme vylepšovat, až dostaneme maximální tok.

Uvažujme, jak by vylepšování mohlo probíhat. Nechť existuje cesta P ze z do s taková, že po všech jejích hranách teče méně, než dovolují kapacity. Pak zjevně můžeme tok upravit tak, aby se jeho velikost zvětšila. Zvolme

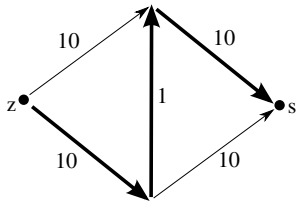
$$\varepsilon := \min_{e \in P} (c(e) - f(e)) .$$

Po každé hraně zvýšíme průtok o ε, čili definujeme nový tok f' takto:

$$f'(e) := \begin{cases} f(e) + \varepsilon & \text{pro } e \in P \\ f(e) & \text{pro } e \notin P \end{cases}$$

To je opět korektní tok: kapacity nepřekročíme (ε jsme zvolili nejvyšší možné, aby se to ještě nestalo) a Kirchhoffovy zákony zůstanou neporušeny, neboť zdroj a stok neomezují a každému jinému vrcholu na cestě P se přítok $f^+(v)$ i odtok $f^-(v)$ zvětší přesně o ε.

Opakujme tento proces tak dlouho, dokud existují cesty, po nichž můžeme tok zlepšovat. Až se algoritmus zastaví (což by obecně nemusel, ale to nás ještě chvíli trápit nemusí), získáme maximální tok? Překvapivě ne vždy. Uvažujme například síť nakreslenou pod tímto odstavcem. Najdeme-li nejdříve cestu přes svislou hranu (na obrázku tučně, zlepšujeme o 1), potom jednu cestu po horní dvojici hran (zlepšujeme o 9) a jednu po spodní dvojici (zlepšujeme také o 9), dostaneme tok o velikosti 19 a žádná další cesta ho už nemůže zlepšit. Ovšem maximální tok v této síti má evidentně velikost 20.



Čísla představují kapacity jednotlivých hran.

Zde by ovšem situaci zachránilo, kdybychom poslali tok velikosti 1 proti směru prostřední hrany – to můžeme udělat třeba odečtením jedničky od toku po směru hrany. Rozšíříme tedy náš algoritmus tak, aby uměl posílat tok i proti směru hran. O kolik můžeme tok hranou zlepšit (ať už přičtením po směru nebo odečtením proti směru) nám bude říkat její *rezerva:*

Definice: *Rezerva hrany* uv je $r(uv) := c(uv) - f(uv) + f(vu)$.

Definice: Hraně budeme říkat *nasycená,* pokud má nulovou rezervu. *Nenasycená cesta* je taková, jejíž všechny hrany mají nenulovou rezervu.

Budeme tedy opakovaně hledat nenasycené cesty a tok po nich zlepšovat. Postupně dokážeme, že tento postup je konečný a že v každé síti najde maximální tok.

Algoritmus FordFulkerson
Vstup: Síť.

1. $f \leftarrow$ libovolný tok, např. všude nulový.
2. Dokud existuje nenasycená cesta P ze z do s, opakujeme:
3. $\quad \varepsilon \leftarrow \min\{r(e) \mid e \in P\}$.
4. $\quad$ Pro všechny hrany $uv \in P$:
5. $\quad\quad \delta \leftarrow \min\{f(vu), \varepsilon\}$
6. $\quad\quad f(vu) \leftarrow f(vu) - \delta$
7. $\quad\quad f(uv) \leftarrow f(uv) + \varepsilon - \delta$

Výstup: Maximální tok f.

Konečnost: Zastaví se Fordův-Fulkersonův algoritmus?

- Pakliže jsou všechny kapacity celá čísla, velikost toku se v každém kroku zvětší alespoň o 1. Algoritmus se tedy zastaví po nejvíce tolika krocích, kolik je nějaká horní mez pro velikost maximálního toku – např. součet kapacit všech hran vedoucích do stoku ($c^+(s)$).
- Pro racionální kapacity využijeme jednoduchý trik. Nechť M je nejmenší společný násobek jmenovatelů všech kapacit. Spustíme-li algoritmus na síť s kapacitami $c'(e) = c(e) \cdot M$, bude se rozhodovat stejně jako v původní síti, protože bude stále platit $f'(e) = f(e) \cdot M$. Nová síť je přitom celočíselná, takže se algoritmus jistě zastaví.
- Na síti s iracionálními kapacitami se algoritmus chová mnohdy divoce, nemusí se zastavit, ba ani konvergovat ke správnému výsledku (viz cvičení 2).

Maximalita: Už víme, že algoritmus se zastaví a vydá jako výsledek nějaký tok f. Je tento tok maximální? Dokážeme to pomocí řezů.

Definice: Pro libovolné dvě množiny vrcholů A a B budeme značit $E(A, B)$ množinu hran vedoucích z A do B, tedy $E(A, B) = E \cap (A \times B)$. Je-li dále f nějaká funkce přiřazující hranám čísla, označíme:

- $f(A, B) := \sum_{e \in E(A,B)} f(e)$ (průtok z A do B)

- $f^\Delta(A,B) := f(A,B) - f(B,A)$ (čistý průtok z A do B)

Definice: *Řez* je uspořádaná dvojice množin vrcholů (A,B) taková, že A a B jsou disjunktní, dohromady obsahují všechny vrcholy a navíc A obsahuje zdroj a B obsahuje stok. Množině A budeme říkat *levá množina řezu,* množině B *pravá. Kapacitu řezu* definujeme jako součet kapacit hran zleva doprava, tedy $c(A,B)$.

Lemma: Pro každý řez (A,B) a každý tok f platí $f^\Delta(A,B) = |f|$.
Důkaz: Opět šikovným sečtením přebytků vrcholů:

$$f^\Delta(A,B) = \sum_{v \in B} f^\Delta(v) = f^\Delta(s).$$

První rovnost získáme počítáním přes hrany: každá hrana vedoucí z vrcholu v B do jiného vrcholu v B přispěje jednou kladně a jednou záporně; hrany ležící celé mimo B nepřispějí vůbec; hrany s jedním koncem v B a druhým mimo přispějí jednou, přičemž znaménko se bude lišit podle toho, který konec je v B. Druhá rovnost je snadná: všechny vrcholy v B kromě spotřebiče mají podle Kirchhoffova zákona nulový přebytek (zdroj totiž v B neleží). ♡

Poznámka: Původní definice velikosti toku coby přebytku spotřebiče je speciálním případem předchozího lemmatu – měří totiž průtok přes řez $(V \setminus \{s\}, \{s\})$.

Důsledek: Pro každý tok f a každý řez (A,B) platí $|f| \leq c(A,B)$. (Velikost každého toku je shora omezena kapacitou každého řezu.)
Důkaz: $|f| = f^\Delta(A,B) = f(A,B) - f(B,A) \leq f(A,B) \leq c(A,B)$. ♡

Důsledek: Pokud $|f| = c(A,B)$, pak je tok f maximální a řez (A,B) minimální. Jinými slovy pokud najdeme k nějakému toku stejně velký řez, můžeme řez použít jako certifikát maximality toku a tok jako certifikát minimality řezu. Následující věta nám zaručí, že je to vždy možné:

Věta: Pokud se Fordův-Fulkersonův algoritmus zastaví, vydá maximální tok.
Důkaz: Nechť se algoritmus zastaví. Uvažme množiny vrcholů $A := \{v \in V \mid$ existuje nenasycená cesta ze z do $v\}$ a $B := V \setminus A$.

Dvojice (A,B) je řez, neboť $z \in A$ (ze z do z existuje cesta délky 0) a $s \in B$ (kdyby s neleželo v B, musela by existovat nenasycená cesta ze z do s, tudíž by algoritmus neskončil, nýbrž by po této cestě stávající tok vylepšil).

Dále víme, že všechny hrany řezu mají nulovou rezervu: kdyby totiž pro nějaké $u \in A$ a $v \in B$ měla hrana uv rezervu nenulovou (nebyla nasycená), spojením nenasycené cesty ze zdroje do u s touto hranou by vznikla nenasycená cesta ze zdroje do v, takže vrchol v by také musel ležet v A, což není možné.

Proto po všech hranách řezu vedoucích z A do B teče tolik, kolik jsou kapacity těchto hran, a po hranách vedoucích z B do A neteče nic. Nalezli jsme tedy řez (A,B) pro nějž $f^\Delta(A,B) = c(A,B)$. To znamená, že tento řez je minimální a tok f maximální. ♡

Tím jsme dokázali větu o správnosti Fordova-Fulkersonova algoritmu:

Věta: Pro každou síť s racionálními kapacitami se Fordův-Fulkersonův algoritmus zastaví a vydá maximální tok a minimální řez.

Důsledek: Síť s celočíselnými kapacitami má aspoň jeden z maximálních toků celočíselný a Fordův-Fulkersonův algoritmus takový tok najde.
Důkaz: Když dostane Fordův-Fulkersonův algoritmus celočíselnou síť, najde v ní maximální tok a ten bude zase celočíselný (algoritmus nikde nevytváří z celých čísel necelá). ♡

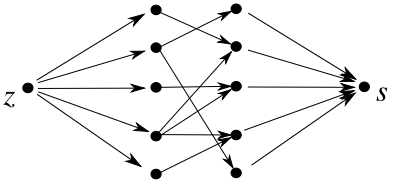
To, že umíme najít celočíselné řešení, není vůbec samozřejmé. (V kapitole o **NP**-úplnosti se setkáme s problémy, které jsou pro racionální čísla snadné a pro celá obtížné.) Ukažme rovnou jednu aplikaci, která celočíselnosti využije.

Hledání největšího párování v bipartitních grafech

Definice: Množina hran $F \subseteq E$ se nazývá *párování*, jestliže žádné dvě hrany této množiny nemají společný vrchol. *Velikostí* párování myslíme počet jeho hran.

Chceme-li v daném bipartitním grafu (V, E) nalézt největší párování, přetvoříme graf nejprve na síť (V', E', c, z, s) takto:

- Nalezneme partity grafu, budeme jim říkat *levá* a *pravá.*
- Všechny hrany zorientujeme zleva doprava.
- Přidáme zdroj z a vedeme z něj hrany do všech vrcholů levé partity.
- Přidáme spotřebič s a vedeme do něj hrany ze všech vrcholů pravé partity.
- Všem hranám nastavíme jednotkovou kapacitu.



Hledání největšího párování v bipartitním grafu.

Nyní v této síti najdeme maximální celočíselný tok. Jelikož všechny hrany mají kapacitu 1, musí po každé hraně téci buď 0 nebo 1. Do výsledného párování vložíme právě ty hrany původního grafu, po kterých teče 1.

Dostaneme opravdu párování? Kdybychom nedostali, znamenalo by to, že nějaké dvě hrany mají společný vrchol. Ovšem kdyby se setkaly ve vrcholu z pravé partity, přitekly by do tohoto vrcholu alespoň 2 jednotky toku a ty by neměly kam odtéci. Analogicky pokud by se setkaly nalevo, musely by z vrcholu odtéci alespoň 2 jednotky, ale ty se tam nemají kudy dostat.

Zbývá nahlédnout, že nalezené párování je největší možné. K tomu si stačí všimnout, že z toku vytvoříme párování o tolika hranách, kolik je velikost toku, a naopak z každého párování umíme vytvořit celočíselný tok odpovídající velikosti. Nalezli jsme bijekci mezi množinou všech celočíselných toků a množinou všech párování a tato bijekce zachovává velikost. Největší tok tedy musí odpovídat největšímu párování.

Navíc dokážeme, že Fordův-Fulkersonův algoritmus na sítích tohoto druhu pracuje překvapivě rychle:

Věta: Pro síť, jejíž všechny kapacity jsou jednotkové, nalezne Fordův-Fulkersonův algoritmus maximální tok v čase $\mathcal{O}(mn)$.
Důkaz: Jedna iterace algoritmu běží v čase $\mathcal{O}(m)$: nenasycenou cestu najdeme prohledáním grafu do šířky, samotné zlepšení toku zvládneme v čase lineárním s délkou cesty. Jelikož každá iterace zlepší tok alespoň o 1,[⟨2⟩] počet iterací je omezen velikostí maximálního toku, což je nejvýše n (uvažujte řez tvořený hranami okolo zdroje). ♡

Důsledek: Největší párování v bipartitním grafu lze nalézt v čase $\mathcal{O}(mn)$.
Důkaz: Předvedená konstrukce vytvoří z grafu síť o $n' = n + 2$ vrcholech a $m' = m + 2n$ hranách a spotřebuje na to čas $\mathcal{O}(m' + n')$. Pak nalezneme maximální celočíselný tok Fordovým-Fulkersonovým algoritmem, což trvá $\mathcal{O}(m'n')$. Nakonec tok v lineárním čase přeložíme na párování. Vše dohromady trvá $\mathcal{O}(m'n') = \mathcal{O}(mn)$. ♡

Cvičení

1. Najděte příklad sítě s nejvýše 10 vrcholy a 10 hranami, na níž Fordův-Fulkersonův algoritmus provede více než milion iterací.
2.** Najděte síť s reálnými kapacitami, na níž Fordův-Fulkersonův algoritmus nedoběhne.
3. Navrhněte algoritmus, který pro zadaný orientovaný graf a jeho vrcholy u a v nalezne největší možný systém hranově disjunktních cest z u do v.
4. Upravte algoritmus z předchozího cvičení, aby nalezené cesty byly vrcholově disjunktní (až na krajní vrcholy).
5. Mějme šachovnici $R \times S$, z níž políčkožrout sežral některá políčka. Chceme na ni rozestavět co nejvíce šachových věží tak, aby se navzájem neohrožovaly. Věž můžeme postavit na libovolné nesežrané políčko a ohrožuje všechny věže v témže řádku i sloupci. Navrhněte efektivní algoritmus, který takové rozestavění najde.
6.* Situace stejná jako v minulém cvičení, ale dvě věže se neohrožují přes sežraná políčka.
7. Opět šachovnice po zásahu políčkožrouta. Chceme na nesežraná políčka rozmístit kostky velikosti 1×2 políčka tak, aby každé nesežrané políčko bylo pokryto právě jednou kostkou.
8. Dopravní problém: Uvažujme továrny $T_1, \ldots, T_p$ a obchody $O_1, \ldots, O_q$. Všichni vyrábějí a prodavají tentýž druh zboží. Továrna T_i ho denně vyprodukuje t_i kusů, obchod O_j denně spotřebuje o_j kusů. Navíc známe bipartitní graf určující, která továrna může dodávat zboží kterému obchodu. Najděte efektivní algoritmus, který zjistí, zda je požadavky obchodů možné splnit, aniž by se překročily výrobní kapacity továren, a pokud je to možné, vypíše, ze které továrny se má přepravit kolik zboží do kterého obchodu.
9.** Uvažujeme o vybudování dolů $D_1, \ldots, D_p$ a továren $T_1, \ldots, T_q$. Vybudování dolu D_i stojí cenu d_i a od té doby důl zadarmo produkuje neomezené množství i-té

[⟨2⟩] Mimochodem, může i o 2, protože při jednotkových kapacitách mohou rezervy být až dvojky.

suroviny. Továrna T_j potřebuje ke své činnosti zadanou množinu surovin (přiřazení surovin továrnám je dáno jako bipartitní graf na vstupu) a pokud jsou v provozu všechny doly produkující tyto suroviny, vyděláme na továrně zisk t_j. Vymyslete algoritmus, který spočítá, které doly postavit, abychom po odečtení nákladů na doly vydělali co nejvíce.

10. Jiná obvyklá definice řezu říká, že řez je množina hran grafu, po jejímž odebrání se graf rozpadne na více komponent (respektive máme-li určený zdroj a stok, skončí tyto v různých komponentách). Srovnejme tuto definici s naší. Množiny hran určené našimi řezy splňují i tuto definici a říká se jim *elementární řezy.* Ukažte, že existují i jiné než elementární řezy. Také ukažte, že všechny minimální řezy jsou elementární.