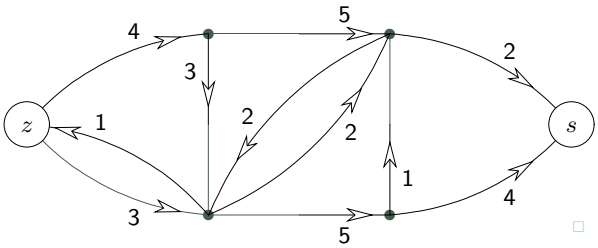
4 Toky v sítích

Nyní se podíváme ještě na jednu oblast úloh, kde našla teorie grafů bohaté uplatnění.Jde o oblast tzv. „síťových“ úloh, kde se jedná (třeba) o potrubní sítě přepravující vodu nebo plyn, o dopravní síť silnic s přepravou zboží, nebo o datovou (počítačovou) síť.

Obvykle nás zajímá problém přenést z daného *zdroje* do daného cíle čili *stoku* co nejvíce substance, za omezujících podmínek kapacit jednotlivých přepravních cest.



Stručný přehled lekce

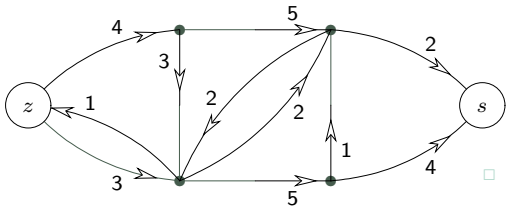
- Definice sítě (ohodnoceného orientovaného grafu) a toku v ní.
- Algoritmus nenasycených cest; jeho vylepšené verze.
- Důsledky pro souvislost, párování a výběr reprezentantů, apod.

4.1 Definice sítě a toku

Základní strukturou pro reprezentaci sítí je orientovaný graf. Vrcholy grafu modelují jednotlivé uzly sítě a hrany jejich spojnice.

Definice 4.1. Síť je čtveřice $S = (G, z, s, w)$, kde

- G je orientovaný graf,
- vrcholy $z \in V(G)$, $s \in V(G)$ jsou *zdroj* a *stok*,
- $w : E(G) \to \boldsymbol{R}^+$ je kladné ohodnocení hran, zvané *kapacita hran*.



Poznámka: V praxi může být zdrojů a stoků více, ale v definici stačí pouze jeden zdroj a stok, z něhož / do nějž vedou hrany do ostatních zdrojů / stoků. (Dokonce pak různé zdroje a stoky mohou mít své kapacity.)

Značení: Pro jednoduchost píšeme ve výrazech znak $e \to v$ pro hranu e přicházející do vrcholu v a $e \leftarrow v$ pro hranu e vycházející z v. □

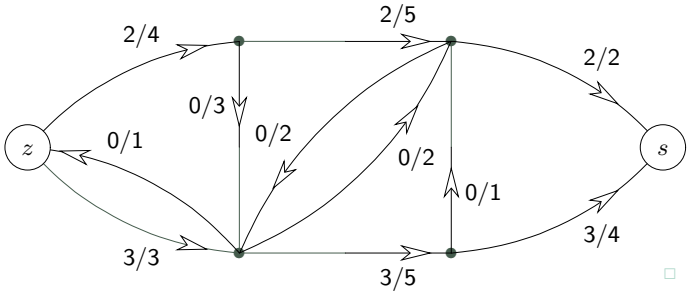
Definice 4.2. Tok v síti $S = (G, z, s, w)$
je funkce $f : E(G) \to \boldsymbol{R}_0^+$ splňující

- $\forall e \in E(G) : \ 0 \leq f(e) \leq w(e),$ (respektování kapacity)
- $\forall v \in V(G), v \neq z, s : \ \sum_{e \to v} f(e) = \sum_{e \leftarrow v} f(e).$ (zachování substance) □

Velikost toku f je dána výrazem $\|f\| = \sum_{e \leftarrow z} f(e) - \sum_{e \to z} f(e).$ □

Značení: Tok a kapacitu hran v obrázku sítě budeme zjednodušeně zapisovat ve formátu F/C, kde F je hodnota toku na hraně a C je její kapacita.

Neformálně tok znamená, kolik substance je každou hranou zrovna přenášeno. Tok je pochopitelně nezáporný a dosahuje nejvýše dané kapacity hrany.



Ve vyobrazeném příkladě vede ze zdroje vlevo do stoku vpravo tok o celkové velikosti 5. □

Poznámka: Obdobně se dá velikost toku definovat u stoku, neboť

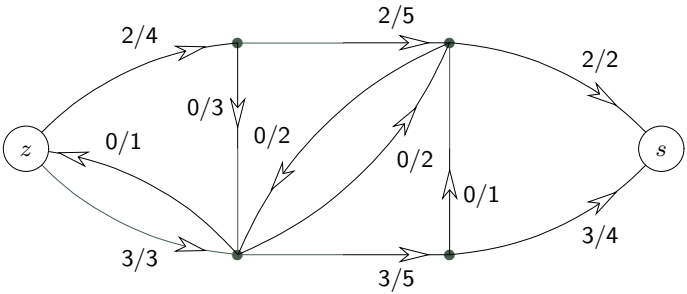
$$0 = \sum_{e \in E} \big(f(e) - f(e)\big) = \sum_{v \in V} \left(\sum_{e \leftarrow v} f(e) - \sum_{e \rightarrow v} f(e) \right) = \sum_{v = z,s} \left(\sum_{e \leftarrow v} f(e) - \sum_{e \rightarrow v} f(e) \right) .$$

(Sčítance uprostřed předchozího vztahu nabývají nulové hodnoty pro všechny vrcholy v kromě z a s dle definice toku.) Proto velikost toku počítaná u zdroje je rovna opačné velikosti toku počítaného u stoku.

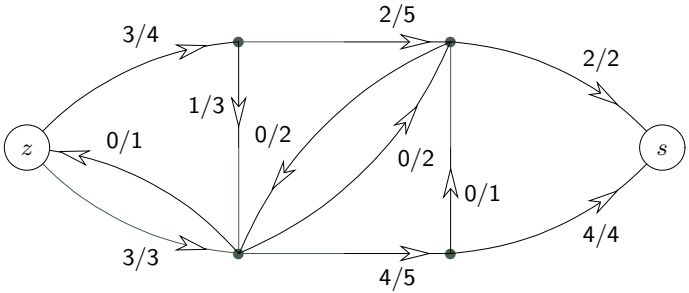
4.2 Hledání maximálního toku

Naším úkolem je najít co největší tok v dané síti. Pro jeho nalezení existují jednoduché a velmi rychlé algoritmy.

Problém 4.3. Maximálního toku v síti
Je dána síť $S = (G, z, s, w)$ a naším úkolem je pro ni najít co největší tok ze zdroje z do stoku s vzhledem k ohodnocení w. □



Tok velikosti 5 uvedený v ukázce v předchozí části nebyl optimální, neboť v této síti najdeme i tok velikosti 6:



Jak však poznáme, že větší tok již v dané síti neexistuje? V této konkrétní ukázce to není obtížné, vidíme totiž, že obě dvě hrany přicházející do stoku mají součet kapacit $2 + 4 + 6$, takže více než 6 do stoku ani přitéct nemůže. □

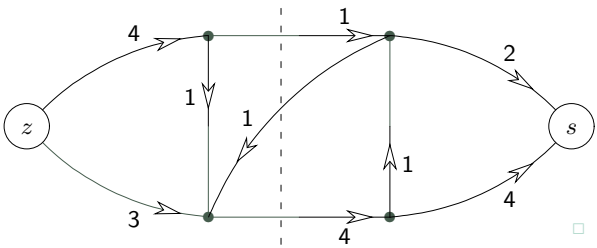
V obecnosti lze použít obdobnou úvahu, kdy najdeme podmnožinu hran, které nelze tokem „obejít“ a které v součtu kapacit dají velikost našeho toku. Existuje však taková množina hran vždy? Odpověď nám dá následující definice a věta.

Pojem řezu v síti

Definice 4.4. Řez v síti $S = (G, z, s, w)$
je podmnožina hran $C \subset E(G)$ taková, že v podgrafu $G - C$ (tj. po odebrání hran C z G) nezbude žádná orientovaná cesta ze z do s.
Velikostí řezu C rozumíme součet kapacit hran z C, tj. $\|C\| = \sum_{e \in C} w(e)$. □

Věta 4.5. *Maximální velikost toku v síti je rovna minimální velikosti řezu.*

Na následujícím obrázku vidíme trochu jinou síť s ukázkou netriviálního minimálního řezu velikosti 5, naznačeného svislou čárkovanou čarou. Všimněte si dobře, že definice řezu mluví o přerušení *všech orientovaných cest ze z do s*, takže do řezu stačí započítat hrany jdoucí přes svislou čáru od z do s, ale ne hranu jdoucí zpět. Proto je velikost vyznačeného řezu $1 + 4 = 5$.



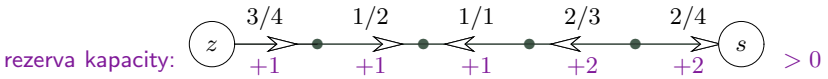
□

Nenasycené cesty v síti

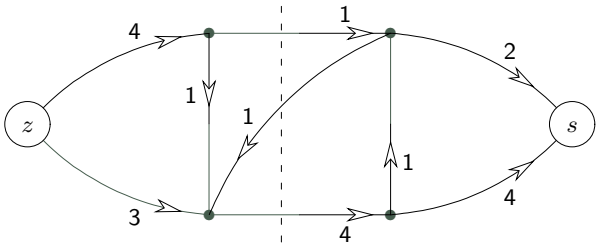
Definice: Mějme síť S a v ní tok f. *Nenasycená cesta* (v S vzhledem k f) je neorientovaná cesta v G z vrcholu u do vrcholu v (obvykle ze z do s), tj. posloupnost navazujících hran $e_1, e_2, \ldots, e_m$, kde $f(e_i) < w(e_i)$ pro e_i ve směru z u do v a $f(e_i) > 0$ pro e_i v opačném směru. □

Hodnotě $w(e_i) - f(e_i)$ pro hrany e_i ve směru z u do v a hodnotě $f(e_i)$ pro hrany e_i v opačném směru říkáme *rezerva kapacity* hran e_i. Nenasycená cesta je tudíž cesta s kladnými rezervami kapacit všech hran. □

Zde vidíme příklad nenasycené cesty ze zdroje do stoku s minimální rezervou kapacity $+1$.



Všimněte si dobře, že cesta není orientovaná, takže hrany na ní jsou v obou směrech.

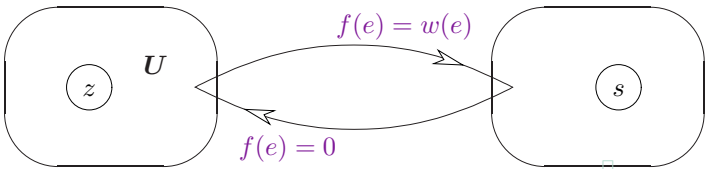


Algoritmus 4.7. Ford–Fulkersonův pro tok v síti.

vstup *síť* $S = (G, z, s, w)$;
tok $f \equiv 0$;
repeat {
    Prohledáváním grafu najdeme množinu U *vrcholů* G,
     do kterých se dostaneme ze z *po nenasycených cestách*;
    if ($s \in U$) {
        P = *(výše nalezená) nenasycená cesta v* S *ze* z *do* s;
        Zvětšíme tok f *o minimální rezervu kapacity hran v* P;
    }
until ($s \notin U$);
výstup *vypíšeme maximální tok* f;
výstup *vypíšeme min. řez jako množinu hran vedoucích z* U *do* $V(G) - U$.

Důkaz správnosti Algoritmu 4.7:
Pro každý tok f a každý řez C v síti S platí $\|f\| \leq \|C\|$. Jestliže po zastavení algoritmu s tokem f nalezneme v síti S řez o stejné velikosti $\|C\| = \|f\|$, je jasné, že jsme našli maximální možný tok v síti S. (Pozor, zastavení algoritmu jsme zatím nezdůvodnili.) □

Takže dokažme, že po zastavení algoritmu nastane rovnost $\|f\| = \|C\|$, kde C je vypsaný řez mezi U a zbytkem grafu G. Vezměme tok f v S bez nenasycené cesty ze z do s. Pak množina U z algoritmu neobsahuje s. Schematicky vypadá situace takto:



Jelikož z U žádné nenasycené cesty dále nevedou, má každá hrana $e \leftarrow U$ (odch. z U) plný tok $f(e) = w(e)$ a každá hrana $e \rightarrow U$ (přich. do U) tok $f(e) = 0$, takže

$$\sum_{e \leftarrow U} f(e) - \sum_{e \rightarrow U} f(e) = \sum_{e \leftarrow U} f(e) = \sum_{e \in C} w(e) = \|C\| \ .$$

To je přesně, co jsme chtěli dokázat o výsledném toku. Zbývá nahlédnout, že $\|C\| = \sum_{e \leftarrow U} f(e) - \sum_{e \rightarrow U} f(e) = \|f\|$, a důkaz je hotov. □

Z důkazu Algoritmu 4.7 odvodíme několik zajímavých faktů:

Fakt: Pokud zajistíme, že Algoritmus 4.7 vždy skončí, dokážeme i platnost Věty 4.5. □

Fakt: Pro celočíselné kapacity hran sítě S Algoritmus 4.7 vždy skončí.

Důsledek 4.8. *Pokud jsou kapacity hran celočíselné, opt. tok také vyjde celočíselně.* □

Vylepšení algoritmu nenasycených cest

Bohužel pro reálná čísla kapacit hran není skončení Algoritmu 4.7 vůbec zaručeno, dokonce se dají najít extrémní případy, které nepovedou k řešení ani v limitě. Proto je potřebné základní algoritmus (i pro potřeby teorie) poněkud vylepšit. □

Algoritmus 4.9. Edmonds–Karpův pro tok v síti
V Algoritmu 4.7 vždy sytíme nejkratší nenasycenou cestu, neboli prohl. naši síť do šířky (viz množina U).

Tato implementace zaručeně skončí po $O(|V(G)| \cdot |E(G)|)$ iteracích cyklu, celkem tedy v čase $O(|V(G)| \cdot |E(G)|^2)$.

Ještě lepších výsledků dosahují následující "chytré" algoritmy.

Algoritmus 4.10. Dinicův pro tok v síti (náznak)
V intencích Algoritmu 4.7 provádíme následující iterace:

- Prohledáváním sítě do šířky nalezneme všechny nenasycené cesty nejkratší délky souběžně, které nám vytvoří "vrstvenou síť" (vrstvy odpovídají nenasycené vzdálenosti vrcholů od zdroje).
- Nalezené nenasycené cesty pak nasytíme novým prohledáním vrstvené sítě.

Tato implementace zaručeně skončí už po $O(|V(G)|)$ iteracích cyklu, ale jednotlivé iterace jsou poněkud náročnější, $O(|V(G)| \cdot |E(G)|)$. □

Algoritmus 4.11. MPM („Tří Indů") pro tok v síti (náznak)
Postupuje se stejně jako v Algoritmu 4.10, jen nesycení v druhém bodě proběhne rychlejším algoritmem v čase $O(|V(G)|^2)$ v každé iteraci, celkem tedy v $O(|V(G)|^3)$.

4.3 Zobecnění definice sítí

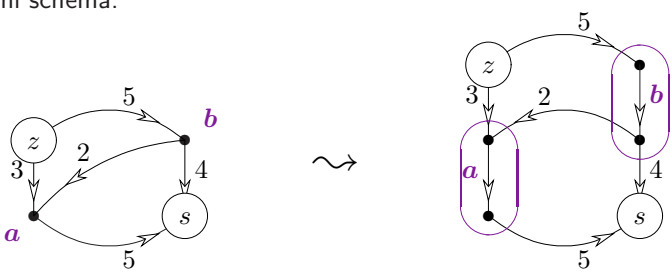
Pojmy sítě a toků v ní lze zobecnit několika směry. My si zde stručně uvedeme tři možnosti.

Sítě s kapacitami vrcholů

U sítě můžeme zadat i *kapacity vrcholů*, neboli kapacitní váhová funkce je dána jako $w : E(G) \cup V(G) \to \boldsymbol{R}^+$.

Význam pro přípustné toky v takové síti je, že žádným vrcholem nemůže celkem „protéct“ více než povolené množství substance. □

Fakt. Takovou síť „zdvojením“ vrcholů snadno převedeme na běžnou síť, ve které kapacity původních vrcholů budou uvedeny u nových hran spojujících zdvojené vrcholy. Viz neformální schéma:



Sítě s dolními kapacitami

Pro hrany sítě lze zadat také jejich *minimální kapacity*, tedy dolní meze příp. toku.

To je modelováno druhou (vedle f) váhovou funkcí $\ell : E(G) \to \boldsymbol{R}_0^+$. Přípustný tok pak musí splňovat $\ell(e) \leq f(e) \leq w(e)$ pro všechny hrany e. □

V této modifikaci úlohy již přípustný tok nemusí vůbec existovat.

Algoritmus 4.12. Tok v síti s dolními kapacitami

I tuto úlohu lze řešit dosud uvedenými nástroji, pokud postupujeme ve dvou fázích: □

- *Nejprve nalezneme přípustnou cirkulaci v síti vzniklé přidáním „zpětné" hrany sz. Toho lze dosáhnout hledáním toku ve speciální síti vyjadřující „přebytky" (či nedostatky) dolních mezí toku v jednotlivých vrcholech.*
- *Poté z přípustné cirkulace jako výchozího stavu už získáme maximální tok kterýmkoliv algoritmem pro toky.* □

Tzv. vícekomoditní toky

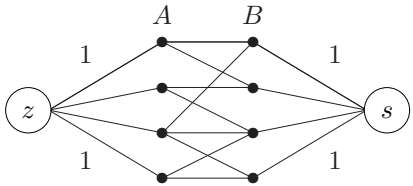
V síti lze najednou přepravovat více typů substancí. To vede na problém tzv. *vícekomoditních toků* v síti. Tento problém je složitější, už není v obecnosti snadno řešitelný a přesahuje rámec našeho textu, takže se jím nebudeme zabývat.

4.4 Speciální aplikace sítí

Bipartitní párování

Definice: *Párování* v (nyní biparitním) grafu G je podmnožina hran $M \subset E(G)$ taková, že žádné dvě hrany z M nesdílejí koncový vrchol. □

Algoritmus 4.13. Nalezení bipartitního párování
Pro daný bipartitní graf G s vrcholy rozdělenými do množin A, B sestrojíme síť S následovně:



Všechny hrany sítě S orientujeme od zdroje do stoku a přiřadíme jim kapacity 1. Nyní najdeme (celočíselný) maximální tok v S Algoritmem 4.7. Do párování vložíme ty hrany grafu G, které mají nenulový tok. □

Důkaz správnosti Algoritmu 4.13: Podle Důsledku 4.8 bude maximální tok celočíselný, a proto každou hranou poteče buď 0 nebo 1. Tím budou vybrány hrany párování a podle zadaných kapacit nebudou sdílet koncové vrcholy. □

Vyšší grafová souvislost

Představme si, že na libovolném grafu G definujeme zobecněnou síť tak, že kapacity všech hran a všech vrcholů položíme rovny 1 v obou směrech. Pak máme:

Důsledek 4.14. *Nechť u, v jsou dva vrcholy grafu G a $k > 0$ je přirozené číslo. Pak mezi vrcholy u a v existuje v G aspoň k disjunktních cest, právě když po odebrání libovolných $k - 1$ vrcholů různých od u, v z G zůstanou u a v ve stejné komponentě souvislosti zbylého grafu.* □

Použitím tohoto tvrzení pro všechny dvojice vrcholů grafu snadno dokážeme dříve uvedenou důležitou Větu 2.7 (Mengerovu).

Výběr různých reprezentantů

Definice: Nechť $M_1, M_2, \ldots, M_k$ jsou neprázdné množiny. *Systémem různých reprezentantů* množin $M_1, M_2, \ldots, M_k$ nazýváme takovou posloupnost různých prvků $(x_1, x_2, \ldots, x_k)$, že $x_i \in M_i$ pro $i = 1, 2, \ldots, k$. □

Věta 4.15. (Hall) *Nechť $M_1, M_2, \ldots, M_k$ jsou neprázdné množiny. Pro tyto množiny existuje systém různých reprezentantů, právě když platí*

$$\forall J \subset \{1, 2, \ldots, k\} : \left|\bigcup_{j \in J} M_j\right| \geq |J|,$$

neboli pokud sjednocení libovolné skupiny z těchto množin má alespoň tolik prvků, kolik množin je sjednoceno.

Důkaz lze podat konstrukcí vhodné sítě podobné té v Algoritmu 4.13: □

- Speciální vrcholy u a v odpovídají zdroji a stoku,
- další vrcholy reprezentují prvky a množiny naší úlohy, □
- ostatní hrany přicházející do vrcholu x_j znázorňují všechny z daných množin, které obsahují prvek x_j.

Poznámka: Tento důkaz nám také dává návod, jak systém různých reprezentantů pro dané množiny nalézt – stačí použít Algoritmus 4.7 na vhodně odvozenou síť.

Segmentace obrazu

Jedním ze základních problémů počítačového vidění je segmentace obrazu na popředí a pozadí.

K jeho řešení lze (mezi mnoha jinými přístupy) zvolit i následující postup:

- Vstupem je matice obrazových bodů (pixelů), z nichž každý má přiřazeny hodnoty pravděpodobnosti bytí v popředí a bytí v pozadí. Dále jsou určeny „separační penalizace“ pro dvojice sousedních pixelů. □
- Síť zkonstruhujeme přidáním univerzálního zdroje z a stoku s, přičemž hrany ze z do pixelů mají kapacitu rovnou pravděpodobnosti bytí v popředí a hrany do s obdobně pro pozadí. Hrany mezi sousedními pixely jsou obousměrné s kapacitami rovnými zmíněným penalizacím.
- Minimální řez v této síti poté určuje „přechody“ mezi popředím a pozadím daného obrazu.