2. Dinicův algoritmus

(zapsala Markéta Popelová)

Na minulé přednášce jsme si ukázali Fordův-Fulkersonův algoritmus. Tento algoritmus hledal maximální tok tak, že začal s tokem nulovým a postupně ho zvětšoval. Pro každé zvětšení potřeboval v síti najít cestu, na které mají všechny hrany kladnou rezervu (po takovéto cestě můžeme poslat více, než po ní aktuálně teče). Ukázali jsme, že pokud takováto cesta existuje, jde tok vylepšit (zvětšit). Zároveň pokud tok jde vylepšit, pak takováto cesta existuje. Dokázali jsme, že pro racionální kapacity je algoritmus konečný a najde maximální tok.

Fordův-Fulkersonův algoritmus má ovšem značné nevýhody. Funguje pouze pro racionální kapacity a je poměrně pomalý. Nyní si ukážeme jiný algoritmus, který nevylepšuje tok pomocí cest, ale pomocí toků... Budeme k tomu potřebovat síť rezerv.

Definice: *Síť rezerv* k toku f v síti $S = (V, E, z, s, c)$ je síť $R = (S, f) = (V, E, z, s, r)$, kde $r(e)$ je rezerva hrany e v toku f.

Konvence: Pro hranu e značí $\overleftarrow{e}$ hranu opačnou. Např. pokud $e = uv$, tak $\overleftarrow{e} = vu$.

Je důležité si uvědomit, že síť rezerv je závislá jak na původní síti S, tak na nějakém toku f v síti S. Síť rezerv R se pak od sítě S liší pouze kapacitami – síť R má jako kapacitu hrany rezervu hrany v původní síti. Pro připomenutí: rezervu hrany e v síti $S = (V, E, z, s, c)$ s tokem f jsme si definovali jako $r(e) = c(e) - f(e) + f(\overleftarrow{e})$.

Než si ukážeme samotný algoritmus, dokážeme si následující lemma.

Lemma: Pro každý tok f v síti S a pro každý tok g v síti $R = (S, f)$ lze v čase $\mathcal{O}(m)$ nalézt tok f' v síti S takový, že $|f'| = |f| + |g|$.

Důkaz:

Důkaz rozdělíme do tří kroků. V prvním kroku si ukážeme, jak budeme tok f' v síti S konstruovat. V druhém kroku dokážeme, že takto zkonstruované f' je opravdu tok. A nakonec ukážeme, že splňuje požadovanou vlastnost, tedy že jeho velikost je součet velikostí toků f a g.

1. konstrukce f'

Pro každou dvojici hran $e, \overleftarrow{e}$ určíme $f'(e)$ a $f'(\overleftarrow{e})$ následovně:

- Pokud $g(e) = g(\overleftarrow{e}) = 0$, pak nastavme:
  - $f'(e) := f(e)$,
  - $f'(\overleftarrow{e}) := f(\overleftarrow{e})$.
- Pokud $g(e) > 0$ a $g(\overleftarrow{e}) = 0$, pak položme:
  - $\varepsilon := \min(g(e), f(\overleftarrow{e}))$,
  - $f'(e) := f(e) + g(e) - \varepsilon$,
  - $f'(\overleftarrow{e}) := f(\overleftarrow{e}) - \varepsilon$.
- Případ $g(e) = 0$ a $g(\overleftarrow{e}) > 0$ vyřešíme obdobně.

- Pokud $g(e) > 0$ a $g(\overleftarrow{e}) > 0$, pak odečteme od toku g cirkulaci po cyklu tvořeném hranami e a $\overleftarrow{e}$:
  - $\delta := \min(g(e), g(\overleftarrow{e}))$,
  - $g'(e) := g(e) - \delta$,
  - $g'(\overleftarrow{e}) := g(\overleftarrow{e}) - \delta$.

  Tok g' nyní spadá pod některý z předchozích případů, které už umíme vyřešit.

2. f' je tok

1. Nejdříve ověřme první podmínku: $\forall e \in E : 0 \leq f(e) \leq c(e)$. Vezměme libovolnou hranu $e \in E$. Podle toho, co teče po hranách e a $\overleftarrow{e}$ v toku g, jsme rozdělili konstrukci toku na tři případy:
   1. Pokud po hranách e a $\overleftarrow{e}$ netekl žádný tok g, pak jsme nastavili $f'(e) := f(e)$ a $f'(\overleftarrow{e}) := f(\overleftarrow{e})$. Tedy pokud f dodržoval kapacity, tak pro f' musí platit to samé.
   2. Pokud po hraně e tekl tok g nenulový a po opačné nulový, tak jsme zvolili: $f'(e) := f(e) + g(e) - \varepsilon$. Víme, že jsme si ε vybrali tak, že $\varepsilon \leq g(e)$. Proto $f'(e) \geq 0$.
      Teď ověřme, že $f'(e) \leq c(e)$. V případě, že $\varepsilon = g(e)$, tak $f'(e) = f(e) \leq c(e)$. V opačném případě platí, že $\varepsilon = f(\overleftarrow{e})$. Pak ovšem
      $$f'(e) = f(e) + g(e) - f(\overleftarrow{e}) \leq$$
      $$\leq f(e) + [c(e) - f(e) + f(\overleftarrow{e})] - f(\overleftarrow{e}) = c(e).$$
      Využili jsme, že g je tok v síti rezerv, tedy $g(e) \leq c(e) - f(e) + f(\overleftarrow{e})$.
      Pro tok $f'(\overleftarrow{e})$ platí, že $\varepsilon \leq f(\overleftarrow{e})$. Proto $f'(\overleftarrow{e}) = f(\overleftarrow{e}) - \varepsilon \geq 0$. Zároveň $f'(\overleftarrow{e}) \leq f(\overleftarrow{e}) \leq c(\overleftarrow{e})$.
      Tím jsme dokázali, že $f'(e)$ i $f'(\overleftarrow{e})$ dodržují kapacity.
   3. V posledním případě tekl po obou hranách kladný tok g. Menší tok z $g(e)$ a $g(\overleftarrow{e})$ jsme vynulovali a od většího odečetli ten menší. Tok $g'(e)$ a $g'(\overleftarrow{e})$ tedy zůstal korektní a tok f' už konstruujeme podle předchozího případu.
2. Teď musíme ještě dokázat, že nový tok neporušuje Kirchhoffovy zákony:
   $$\forall v \in V \setminus \{z, s\} : f'^{\Delta}(v) = 0.$$
   Vezměme si libovolnou hranu $e = uv \in E$. Uvědomme si, že při přechodu z $f(e)$ na $f'(e)$ a z $f(\overleftarrow{e})$ na $f'(\overleftarrow{e})$ bylo:
   - $f^{\Delta}(u)$ sníženo o $g(e)$
   - $f^{\Delta}(v)$ zvýšeno o $g(e)$.

Sečteme-li úpravy na všech hranách, dostaneme:

$$f'^{\Delta}(v) = f^{\Delta}(v) + \sum_{u:uv\in E} g(uv) - \sum_{u:vu\in E} g(vu) =$$

$$= f^{\Delta}(v) + g^{+}(v) - g^{-}(v) = f^{\Delta}(v) + g^{\Delta}(v).$$

Jelikož f byl tok, tak $f^{\Delta}(v) = 0$ a jelikož g byl tok, tak $g^{\Delta}(v) = 0$. Proto $f'^{\Delta}(v) = f^{\Delta}(v) + g^{\Delta}(v) = 0$.

Tím jsme dokázali, že f' je tok v síti S.

3. $|f'| = |f| + |g|$

Použijme vztah pro součet přebytků z předchozího kroku:

$$|f'| = f'^{\Delta}(s) = f^{\Delta}(s) + g^{\Delta}(s) = |f| + |g|.$$

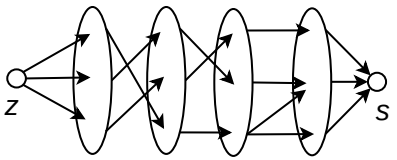
♡

Pro algoritmus budeme potřebovat vybírat kvalitní toky g v síti rezerv. Pokud se nám to bude dařit, bude se tok f' rychle zvětšovat, až bychom mohli dojít k maximálnímu toku. Nejlépe by se nám hodily co největší toky v síti rezerv. Kdybychom si dali za cíl najít vždy maximální tok v síti rezerv, výsledek by byl sice krásný (dostali bychom tak rovnou i maximální tok v původní síti), ale problém hledání maximálního toku bychom pouze přenesli na jinou síť. Naše požadavky na tento tok budou tedy takové, aby byl dostatečně velký, ale abychom během jeho hledání nestrávili moc času. Podívejme se, jak se s tímto problémem vyrovná *Dinicův algoritmus*. Nejdříve si ale zadefinujme několik pojmů.

Definice: Tok f je *blokující*, jestliže pro každou orientovanou cestu P ze z do s existuje hrana $e \in P$ taková, že $f(e) = c(e)$.

Definice: Síť je *vrstevnatá (pročištěná)*, když všechny vrcholy a hrany leží na nejkratších cestách ze z do s.

Dinicův algoritmus začíná s nulovým tokem. Potom vždy podle toku f sestrojí síť rezerv a v ní vymaže hrany s nulovou rezervou. Pokud v této promazané síti rezerv neexistuje cesta ze zdroje do stoku, tak skončí a prohlásí tok f za maximální. Jinak pročistí síť rezerv tak, aby se z ní stala vrstevnatá síť (rozdělí vrcholy do vrstev podle vzdálenosti od zdroje a odstraní přebytečné hrany). Ve vrstevnaté síti najde blokující tok, pomocí něhož zlepší tok f. Pak opět pokračuje sestrojením sítě rezerv na tomto vylepšeném toku f atd.



Pročištěná síť rozdělená do vrstev

Algoritmus (Dinicův)

1. $f \leftarrow$ nulový tok.
2. Sestrojíme síť rezerv R a smažeme $e : r(e) = 0$.
3. $l \leftarrow$ délka nejkratší cesty ze z do s v R.
4. Pokud $l = \infty$, zastavíme se a vrátíme f.
5. Pročistíme síť $R \rightarrow$ síť C.
6. $g \leftarrow$ blokující tok v C.
7. Zlepšíme tok f pomocí g.
8. GOTO 2.

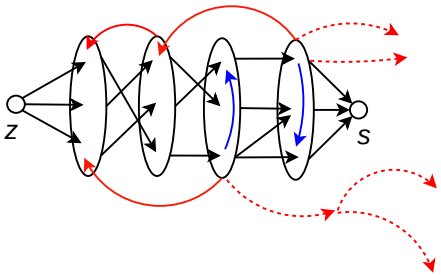
Pozorování: Pokud se algoritmus zastaví, vydá maximální tok.

Důkaz: Víme, že f je stále korektní tok (jediné, jak ho měníme je přičítání toku g, což je, jak jsme si dokázali, „neškodná operace“). Jakmile neexistuje cesta ze z do s v R, tak je f maximální tok, neboť v tuto dobu by se zastavil (a vydal maximální tok) i Fordův-Fulkersonův algoritmus, který je korektní. ♡

A teď ještě musíme ujasnit, jak budeme čistit síť rezerv a vybírat blokující tok g.

Algoritmus pročištění sítě rezerv

1. Rozdělíme vrcholy do vrstev podle vzdálenosti od z.
2. Odstraníme vrstvy za s (tedy vrcholy, které jsou od z vzdálenější než s), hrany do minulých vrstev a hrany uvnitř vrstev.
3. Odstraníme „slepé uličky“, tedy vrcholy s $\deg^{out}(v) = 0$, a to opakovaně pomocí fronty. (Nejdříve zařadíme do fronty všechny vrcholy s $\deg^{out}(v) = 0$. Pak dokud není fronta prázdná, vždy vybereme vrchol v z fronty, odstraníme v a všechny hrany uv. Pro každý takový vrchol u zkontrolujeme, zda se tím nesnížil výstupní stupeň vrcholu u na nulu ($\deg^{out}(u) = 0$). Pokud snížil, tak ho zařadíme do fronty.)



Nepročištěná síť. Obsahuje zpětné hrany, hrany uvnitř vrstvy a slepé uličky.

Hledání blokujícího toku začneme s tokem nulovým. Pak vezmeme vždy orientovanou cestu ze zdroje do stoku v síti C. V této cestě najdeme hranu s nejnižší

hodnotou výrazu $r(e) - g(e)$ (neboli $c(e) - f(e)$ v původní síti). Tuto hodnotu označíme ε. Pak ke všem hranám na této cestě přičteme ε. Pokud tok g na nějaké hraně dosáhne kapacity hrany, což je zde $r(e)$, tak hranu vymažeme. Následně síť dočistíme, aby splňovala podmínky vrstevnaté sítě. A pokud ještě existuje nějaká orientovaná cesta ze zdroje do stoku, tak opět pokračujeme s touto cestou.

Algoritmus hledání blokujícího toku

1. $g \leftarrow$ nulový tok.
2. Dokud existuje orientovaná cesta P ze z do s v C, opakuj:
3. $\quad \varepsilon \leftarrow \min_{e \in P}(r(e) - g(e))$.
4. $\quad$ Pro $\forall e \in P : g(e) \leftarrow g(e) + \varepsilon$.
5. $\quad\quad$ Pokud $g(e) = r(e)$, smažeme e z C.
6. $\quad$ Dočistíme síť zase pomocí fronty.

Časová složitost Rozeberme si jednotlivé kroky algoritmu.

1. Inicializace toku f ... $\mathcal{O}(m)$.
2. Sestrojení sítě rezerv a smazání hran s nulovou rezervou ... $\mathcal{O}(m+n)$.
3. Najití nejkratší cesty (prohledáváním do šířky) ... $\mathcal{O}(m+n)$.
4. Zkontrolování délky nejkratší cesty ... $\mathcal{O}(1)$.
5. Pročištění sítě ... $\mathcal{O}(m+n)$.
    1. Rozdělení vrcholů do vrstev – provedlo již prohledávání do šířky ... $\mathcal{O}(1)$.
    2. Odstranění některých hran ... $\mathcal{O}(m+n)$.
    3. Odstranění „slepých uliček“ pomocí fronty – každou hranu odstraníme nejvýše jedenkrát, každý vrchol se dostane do fronty nejvýše jedenkrát ... $\mathcal{O}(m+n)$.
6. Najití blokujícího toku g ... $\mathcal{O}(m \cdot n)$.
    1. Inicializace toku g ... $\mathcal{O}(m)$.
    2. Najití orientované cesty v pročištěné síti rezerv (stačí vzít libovolnou cestu ze zdroje, neboť každá z nich v této síti vede do stoku) ... $\mathcal{O}(n)$.
    3. Výběr minima z výrazu $r(e) - g(e)$ přes všechny hrany cesty – ta může být dlouhá nejvýše n ... $\mathcal{O}(n)$.
    4. Přepočítání všech hran cesty ... $\mathcal{O}(n)$.
    5. Smazání hran cesty, jejichž tok $g(e)$ se zvýšil na hodnotu $r(e)$ ... $\mathcal{O}(n)$.
    6. Dočišťování vyřešme zvlášť.

    Vnitřní cyklus (kroky 2 až 6) provedeme nejvýše m krát, neboť při každém průchodu vymažeme alespoň jednu hranu (tak jsme si volili ε).
    Čištění během celého hledání blokujícího toku g v pročištěné síti rezerv trvá dohromady $\mathcal{O}(m+n)$, neboť každou hranu a vrchol smažeme nejvýše jedenkrát.

Najití blokujícího toku bude tedy trvat $\mathcal{O}(m \cdot n + (m+n)) = \mathcal{O}(m \cdot n)$.

7. Zlepšení toku f pomocí toku g ... $\mathcal{O}(m)$.
8. Skok na 2. krok ... $\mathcal{O}(1)$.

Zbývá nám jen určit, kolikrát projdeme vnějším cyklem (fází). Dokážeme si lemma, že hodnota l vzroste mezi průchody vnějším cyklem (fázemi) alespoň o 1. Z toho plyne, že vnějším cyklem můžeme projít nejvýše n-krát, neboť cesta v síti na n vrcholech může být dlouhá nejvýše n.

Uvědomme si, že uvnitř vnějšího cyklu převládá člen $\mathcal{O}(m \cdot n)$, takže celková časová složitost bude $\mathcal{O}(n^2 \cdot m)$.

Lemma: Hodnota l (délka nejkratší cesty ze z do s v pročištěné síti) vzroste mezi fázemi alespoň o 1.

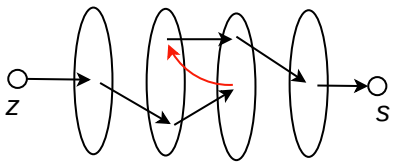
Důkaz:

Podíváme se na průběh jednoho průchodu vnějším cyklem. Délku aktuálně nejkratší cesty ze zdroje do stoku označme l. Všechny původní cesty délky l se během průchodu zaručeně nasytí, protože tok g je blokující. Musíme však dokázat, že nemohou vzniknout žádné nové cesty délky l nebo menší. V síti rezerv totiž mohou hrany nejen ubývat, ale i přibývat: pokud pošleme tok po hraně, po které ještě nic neteklo, tak v protisměru z dosud nulové rezervy vyrobíme nenulovou. Rozmysleme si tedy, jaké hrany mohou přibývat.

Hrany mohou přibývat jen tehdy, když jsme po opačné hraně něco poslali. Ale my něco posíláme po hranách pouze z vrstvy do té následující. Hrany tedy přibývají do minulé vrstvy.

Vznikem nových hran by proto mohly vzniknout nové cesty ze zdroje do stoku, které používají zpětné hrany. Jenže cesta ze zdroje do stoku, která použije zpětnou hranu, musí alespoň jednou skočit o vrstvu zpět a nikdy nemůže skočit o více než jednu vrstvu dopředu, a proto je její délka alespoň $l+2$. Pokud cesta novou zpětnou hranu nepoužije, má buď délku $> l$, což je v pořádku, nebo má délku $= l$, pak je zablokovaná.

Tím je lemma dokázáno. ♡



Cesta užívající novou zpětnou hranu

Všechna dokázaná tvrzení můžeme shrnout do následující věty:

Věta: Dinicův algoritmus najde maximální tok v čase $\mathcal{O}(n^2 \cdot m)$.

Poznámka: Algoritmus se chová hezky na sítích s malými celočíselnými kapacitami, ale kupodivu i na různých jiných sítích. Často se používá, neboť se chová efektivně.

A je mnoho způsobů, jak ho ještě vylepšovat, či odhadovat nižší složitost na speciálních sítích.

Poznámka: Algoritmus nevyžaduje racionální kapacity! Další z důvodů, proč maximální tok existuje i v síti s iracionálními kapacitami.