Toky, cesty, řezy

2. ledna 2013

Obsah

1 Úvod

1.1 Multikomoditní toky

Máme více stoků a zdrojů. Dvě možnosti – chceme maximalizovat jen součet všech toků (součtový) a nebo máme nějaký požadavek na procenta/velikost nebo tak (souběžný). Při snížení počtu komodit na jednu oboje degraduje na obyčejný tok. Lze je řešit pomocí lineárního programování. Řezové problémy k tomu nejsou v přímé dualitě.

Souvisí s reálnými problémy (logistika) a s grafovými parametry (například s expanzí grafu).

Budeme používat lineární programování, semidefinitní programování, geometrii, pravděpodobnostní metodu a vnořování metrických prostorů.

1.2 Toky omezené délky

Normální toky neuvažují délku cesty. Tady si délku nějak omezíme. Také přestane platit dualita, někdy je NP-těžké počítat ten tok, řez vždycky.

1.3 Vícecestné toky

Jsou n-tice hranově disjunktních cest, vždy po celé této n-tici musí téct stejně (odolnost proti výpadkům sítě – aby nešlo jen tak přehlodat jednu klíčovou hranu).

1.4 Celočíselné toky

U těchto speciálních toků již ztrácíme tu hezkou vlastnost, že vždycky na celočíselných kapacitách vychází celočíselné výsledky.

2 Multikomoditní součtové toky

Na vstupu máme graf $G = (V, E)$, je neorientovaný a $c : E \to \mathbb{R}^+$ jsou kapacity hran. Potom jsou komodity $(s_1, t_1), (s_2, t_2), \ldots, (s_k, t_k)$ jsou komodity a dvojice zdroj-spotřebič. Potom $\mathcal{P}_i$ si označíme množinu všech cest z s_i do p_i. Dále vezmeme $\mathcal{P} = \bigcup_{i=1}^{k} \mathcal{P}_i$. Samozřejmě nesmíme překročit kapacitu hran. Když pustím proti sobě různé komodity, potom se sčítají. Tedy, $\sum_{p:e\in p} f_p \leq c(e), f_p \geq 0$. Chceme maximalizovat součet jednotlivých toků.

Můžeme převést na lineární programování tak, že zavedeme pravidla pro vrcholy, pro kapacity hran a fungujeme na jednotlivých komoditách.

2.1 Multikomoditní řež

Chceme najít nějakou množinu hran takovou, že po odebrání nebude žádná dvojice (s_i, t_i) sdílet stejnou komponentu. Snažíme se minimalizovat součet kapacit hran v tomto řezu.

Pro každou hranu máme proměnnou o hodnotě buď $x(e) \in \{0, 1\}$. Chceme minimalizovat $\sum x(e) \cdot c(e)$ a $\forall p \in P; \sum_{e \in p} x(e) \geq 1$. Ale chceme celočíselné řešení. Tyto problémy jsou k sobě skoro duální (po úpravě, že hrana může mít větší hodnotu, než 1). Potřebujeme ale celočíselné řešení – jak moc se liší?

Algoritmus 1 (Algoritmus potvrzující řez):

Vezmu si $\overline{V} = V$ a $e = 1$. Dokud $\overline{V}$ obsahuje nějaké s_i a t_i, tak najdeme nějaké $V_e \subseteq \overline{V}$, která je separuje. Potom z $\overline{V}$ odeberu V_e, k e přičtu 1.

Jinými slovy, toto rozděluje graf na části a vypisujeme všechny hrany, které cestují mezi komponentami.

☺

Vyřešíme duální úlohu lineárního programování. $B_x(s_i, r) := \{v \in V | d_x(s_i, v) \leq r\}$, kde $d_x(s_i, v)$ je délka nejkratší cesty z s_i do v. Objem $V_x(s_i, r) = \frac{\Phi}{k} + \sum_{e=(u,v) \in E, u,v \in B_x(s_i,r)} + \sum$ ze zbytku hran podle toho, jaká délka zbývá (tedy, objem potrubí do vzdálenosti r od s_i). Φ je celkový objem systému. $C_x(s_i, r)$ je součet kapacit přes hrany, které vedou z té koule ven. Fixujme s_i a budeme objem považovat za funkci poloměru. Tato je neklesající. Je po částech spojitá, je nespojitá jen v tolika bodech, kolik máme vrcholů.

Lemma 1 *Existuje poloměr $r < 0.5$ takový, že velikost toho řezu vůči velikosti potrubí je nejvýše $2 \cdot \log 2 \cdot k$.*

K čemu nám toto pomůže? Najdeme dvojici s_i, t_i a aplikujeme lemma, vyndáme správně velkou kouli a pokračujeme. Celkově neodebereme určitě více než 2Φ koulí.

Správné r budeme hledat taková, která jsou mezi vrcholy – ve spojitých částech.

Algoritmus nalezne $4 \cdot \ln 2k$ aproximaci minimálního multikomoditního řezu.

Určitě multitok je vždy $\leq$ multiřez. Opačný odhad lze vzít z předchozího algoritmu.

2.2 Souběžný multitok a multiřez

Máme graf, máme kapacity, máme komodity (každá má zdroj, stok a požadavek, kolik chce protéct). Kromě obvyklých požadavků chceme také, aby byly požadavky dodrženy rovnoměrně (tedy, třeba všechny na 35%).

Tok lze opět vyřešit lineárním programem. Pro podmnožinu vrcholů $S \subseteq V(G)$ označíme $\delta(S)$ množinu všech hran, které vedou ven z S, tedy $\{\{u,v\} \in E(G); |\{u,v\} \cap S| = 1\}$. $I(S)$ jsou indexy komodit, které odebrání S rozpojí.

Pro $S \subseteq V$ je ***hustota*** je:

$$\frac{\sum_{e \in \delta(S)} c(e)}{\sum_{i \in I(S)} d_i}$$

Hledáme nějaký řez s co nejmenší hustotou.

Pozorování 1 *Pro nezáporná čísla $a_1, \ldots, a_n$ a kladná čísla $b_1, \ldots, b_n$ platí:*

$$\min_i \frac{a_i}{b_i} \leq \frac{\sum a_j}{\sum b_j}$$

Když $I \subseteq 1, \ldots, k, D(I) = \sum_{i \in S} D(i)$ budeme značit součet požadavků vybraných komodit. D bude součet všech a $H(D)$ bude D-té harmonické číslo.

Vybereme si množinu indexů, že požavadek té komodity je velký, tedy alespoň $\frac{1}{D(I) \cdot H(D)}$. Poté najdeme nejmenší multiřez pro tyto komodity.

Najdeme ho tak, že seřadíme požadavky dle velikosti a začneme brát od těch největších, pokaždé kontrolovat, jestli to ještě jde.

2.2.1 Lepší algoritmus řezu

Označíme $d(u,v)$ délku nejkratší cesty mezi vrcholy u, v.

Budeme předpakládat, že nejklatší cesta mezi dvěma vrcholy vede po přímé hraně (pokud taková existuje) a požadavek každé komodity je roven délce nejklatší hrany.

To, že toto můžeme udělat, dokážeme sporem, předpokládejme, že máme nějaké optimální řešení, které nesplňuje to první. Vezmeme nejmenší takové, které porušuje co nejméně. Můžeme upravit délku hrany, aniž bychom to porušili. Tím podmínku neporušíme, protože tam byla jiná cesta.

Určitě musí platit, že $y(i) \geq d(s_i, t_i)$, protože jinak to porušuje podmínku, ale pak to můžu přenastavit na rovnost.

Tedy, na to řešení lze nahlížet jako na metrický prostor.

Lemma 2 *Nechť $f : V \to \mathbb{R}^d$ pro nějaké d a nechť $x(e) = |f(u) - f(v)|_{l_1}$ a pro $y(i) = |f(s_1) - s(t_i)|_{l_1}$ a spočítejme $\beta = \sum_{i=1}^{k} d_i \cdot y(i)$. Potom tvrdíme, že $\left(\frac{x}{\beta}, \frac{y}{\beta}\right)$ je řešení pro lineární program duální k tomu tokovému.*

Lemma 3 *Nechť (x, y) je řešení indukované zobrazením $f : V \to \mathbb{R}^d$ (jako v minulém lemmatu). Pak lze v polynomiálním čase najít řez $S \subseteq V$ s hustotou $\rho(S) \leq \sum c(e) \cdot x(e)$.*

Důkaz:
Pro dané f si označme $\mu(u, v) := |f(u) - f(v)|_{l_1}$. Pro každou podmnožinu vrcholů $S \subseteq V$ definujme $\forall u, v \in V \ \mu_s(u, v) \in \{0, 1\}$, podle toho jestli je právě jeden uvnitř. Říká se tomu řezová pseudometrika.

Lemma 4 *Existují konstanty $\lambda_s \geq 0$ pro každou podmnožinu tž $\forall u, v \in V; \mu(u, v) = \sum_{S \subseteq V} \lambda_s \mu_s(u, v)$. Navíc množina nenulových λ_s je nejvýše $n \cdot d$, kde d je dimenze prostoru, n je počet vrcholů.*

Důkaz:
Uvažme příspěvek první souřadnice do μ. Očíslujme vrcholy V podle první souřadnice. Označme $S(l) = \{v_1, \ldots, v_l\}$. Vezmeme vrcholy $v_i, v_j, i < j$.

☻

☻

Lemma 5 *Pro libovolné přípustné řešení (x, y) lze sestrojit zobrazení $f : V \to \mathbb{R}^d$, že indukuje řešení $(\overline{x}, \overline{y})$ s vlastností $\sum c(e) \cdot \overline{x}(e)$, které je nejvýše $O(\log k)$ krát horší, než účelová funkce (x, y).*

Tvrzení 1 *Existuje polynomiální pravděpodobnostní algoritmus, který najde $O(\log k)$ aproximaci nejřidšího řezu.*

Lemma 6 *Pro libovolné přípustné řešení (X, Y) lze v polynomiálním čase sestrojit vnoření $f : V \to \mathbb{R}^d$, které indukuje řešení (X', Y') takové, že hodnota účelové funkce se zhorší max. o $O(\log k)$.*

Předpokládejme, že každému vrcholu dáme jen jednu souřadnici (vnořujeme do $\mathbb{R}^1$) a zobrazíme ho $\min d_x(u, v)$. Vzdálenost tedy bude nejvýše tak velká, jako původní. Když budeme zobrazovat do d dimenzionální, potom nám vyjde maximálně d krát větší.

Nakonec to vyjde, že narostou nejvýše $\log^2 k$ a alespoň $\log k$.

3 Počítání lineárního programování

3.1 Simplexový algoritmus

Máme program, který má exponenciálně proměnných. To mírně komplikuje situaci, potřebujeme si stav lépe pamatovat. Budeme si pamatovat jen ty proměnné, které nejsou 0 a těch bude mnohem méně.

Vyjde to lépe, než si ten program udělat tak, že máme proměnné pro hrany a komodity a podmínky pro každý vrchol (což má polynomiálně mnoho), ale v praxi se tamto minulé lépe „komprimuje“.