ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta aplikovaných věd
Katedra matematiky

Toky v sítích s konvexními cenami
BAKALÁŘSKÁ PRÁCE

2024 Lucie Mänzelová

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta aplikovaných věd
Akademický rok: 2023/2024

ZADÁNÍ BAKALÁŘSKÉ PRÁCE
(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Lucie MÄNZELOVÁ**
Osobní číslo:	**A21B0064P**
Studijní program:	**B0541A170007 Matematika a její aplikace**
Téma práce:	**Toky v sítích s konvexními cenami**
Zadávající katedra:	**Katedra matematiky**

Zásady pro vypracování

1. Seznamte se s problematikou optimalizace toků v sítích, lineárního programování a konvexní optimalizace.
2. Proveďte rešerši metod a algoritmů pro úlohy optimalizace toků.
3. Implementujte vybrané přístupy ve vhodném prostředí.
4. Proveďte analýzu a srovnání škálovatelnosti jednotlivých metod.

Rozsah bakalářské práce: **20-50 stran**
Rozsah grafických prací: **dle potřeby**
Forma zpracování bakalářské práce: **tištěná**

Seznam doporučené literatury:

- Jungnickel: Graphs, Networks and Algorithms, Springer 1999.
- Ahuja, Magnanti, Orlin: Network flows: Theory, Algorithms, and Applications, Pearson 1993.
- Další literatura bude upřesňována průběžně.

Vedoucí bakalářské práce: **Doc. Ing. Roman Čada, Ph.D.**
Katedra matematiky

Datum zadání bakalářské práce: **2. října 2023**
Termín odevzdání bakalářské práce: **22. května 2024**

Západočeská univerzita v Plzni
Fakulta aplikovaných věd
L.S.

Doc. Ing. Miloš Železný, Ph.D.
děkan

Doc. Ing. Marek Brandner, Ph.D.
vedoucí katedry

V Plzni dne 2. října 2023

Čestné prohlášení

Prohlašuji, že jsem bakalářskou práci na téma „Toky v sítích s konvexními cenami“ vypracovala samostatně a výhradně s použitím literatury a pramenů uvedených v seznamu na konci této práce.

V Plzni dne ....

vlastnoruční podpis

Poděkování

Chtěla bych poděkovat vedoucímu práce Doc. Ing. Romanu Čadovi, Ph.D. za odborné vedení, cenné rady, veškerý čas a trpělivost v průběhu zpracování bakalářské práce.

Abstrakt

Tato bakalářská práce je zaměřena na úlohu toků v sítích. Jsou zde uvedeny úlohy hledání maximálního toku a hledání toku s minimální cenou. Hlavním výstupem práce je seznámení se s toky s konvexními cenami, implementace algoritmů řešících tuto úlohu a následné jejich porovnání.

Klíčová slova

toky v sítích, maximální tok, tok s minimální cenou, sítě s konvexními cenami hran, algoritmy

Abstract

This bachelor thesis is focused on the problem of flows in networks. The tasks of finding the maximum flow and finding the minimum cost flow are presented here. The main output of the work is familiarization with flows with convex prices, implementation of algorithms solving this task and their subsequent comparison.

Keywords

flows in networks, maximum flow, flow with minimum cost, networks with convex edge costs, algorithms

Obsah

1 Úvod

V této práci se budeme zabývat problematikou toků v sítích. V úvodní části představíme základní pojmy a definice klíčové pro pochopení následujícího textu. Následně se zaměříme na lineární programování jako nástroj pro formulaci a řešení klasických úloh toků. Detailně se budeme věnovat problémům nalezení maximálního toku a toku s minimální cenou a představíme základní algoritmy, které tyto úlohy řeší.

Dále se budeme zabývat konvexním programováním a dualitou, přičemž zobecníme úlohu hledání toku s minimální cenou (s lineárními cenovými funkcemi na hranách) na úlohu s konvexními cenami. V této souvislosti představíme základní algoritmus po částech lineárních funkcí a Frank-Wolfeův algoritmus, které se využívají pro řešení toků v sítích s konvexními cenami.

Tyto algoritmy implementujeme v prostředí Matlab a otestujeme je na sítích v rozpětí 50-1000 vrcholů. Testovány budou sítě s lineárním počtem hran $m = 8n$ a superlinárním počtem hran $m \approx n\sqrt{n}$. Dále bude také testovaná volba parametru α. Výsledky experimentů budou analyzovány a graficky znázorněny.

2 Základní pojmy a definice

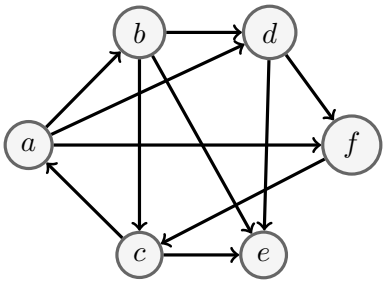
Tato kapitola je zpracována s využitím zdrojů [1],[2], [3] a [4].

Grafy se využívají v mnoha oblastech, zejména v situacích, ve kterých potřebujeme znázornit objekty a jejich vzájemné propojení. Používají se například v počítačových nebo sociálních vědách, při optimalizacích, v ekonomice, ale také ve fyzice či chemii. V této práci budeme využívat orientovaný graf, proto ho nyní formálně definujme.

Definice 2.0.1. *Mějme množinu vrcholů V a množinu hran $E \subseteq V \times V$. Dvojici $D = (V, E)$ pak nazýváme* ***orientovaným grafem****.*

Každý prvek $v \in V$ nazveme vrcholem a každý prvek $e \in E$ nazveme orientovanou hranou (dále pouze hranou). Velikost množiny všech vrcholů grafu D budeme značit $n = |V|$ a velikost množiny všech hran $m = |E|$. Hrana je definovaná uspořádanou dvojicí vrcholů $e = (u, v)$, kde u nazýváme počátečním vrcholem a v koncovým vrcholem hrany. V této práci se zaměříme na orientované grafy bez smyček, což jsou grafy, v nichž každá hrana spojuje dva různé vrcholy.

Každý graf může být reprezentován kresbou v rovině viz obrázek 2.1, kde jsou vrcholy ($V = \{a, b, c, d, e, f\}$) reprezentovány body a hrany spojnicemi vrcholů (např. $E = \{(a, b), (b, d), ...\}$). Pro zakreslení orientace hrany se používá šipka směřující ke koncovému vrcholu.



Obrázek 2.1: Příklad zakreslení grafu do roviny

Pro každý vrchol $v \in V$ definujeme vstupní a výstupní stupeň.

Definice 2.0.2. *Množina všech hran, vedoucích do vrcholu v se nazývá* ***množina vstupních hran vrcholu*** *v a značí se $\delta^{IN}(v)$.*

$$\delta^{IN}(v) = \{e \in E \mid e = (u, v),\ u \in V\}$$

Počet hran v této množině se pak nazývá ***vstupní stupeň vrcholu*** *v.*

Množina všech hran, vedoucích z vrcholu v se nazývá ***množina výstupních hran vrcholu*** *v a značí se $\delta^{OUT}(v)$.*

$$\delta^{OUT}(v) = \{e \in E \mid e = (v, u),\ u \in V\}$$

Počet hran v této množině se pak nazývá ***výstupní stupeň vrcholu*** v.

Dále zaveďme pojmy orientovaná cesta a cyklus, protože s nimi budeme v následujících kapitolách pracovat. K jejich definování využijeme sled, který je jejich zobecněním.

Definice 2.0.3. ***Sledem*** S *v grafu* D *nazveme posloupnost vrcholů* $v_1, v_2, ..., v_k$ *takovou, že existuje hrana* e_i *vedoucí z vrcholu* v_i *do* v_{i+1}*, pro* $i = 1, ...k-1$*. Sled* S *označíme*

$$S = (v_1, v_2, ..., v_{k-1}, v_k)$$

Příkladem sledu na obrázku 2.1 může být $S = (a, b, d, f, c, a, d, f, c, e)$. Cestou v grafu rozumíme sled, ve kterém žádný vrchol není použit vícekrát.

Definice 2.0.4. ***Cestou*** P *v grafu* D *rozumíme sled takový, že platí* $\forall v_i, v_j \in S : \ v_i \neq v_j$*, pro* $i \neq j$.

Speciálním případem cesty je cyklus.

Definice 2.0.5. ***Cyklus*** C *v grafu* D *je cesta, která má koncový vrchol shodný s počátečním.*

Příkladem cyklu na obrázku 2.1 může být $C = (a, f, c, a)$.

2.1 Toky a cirkulace

V této části budeme hovořit obecně o toku jako takovém a pojmech, které s ním souvisí. Konkrétně tedy o divergenci, intenzitě vrcholů, cirkulaci a Kirchhoffovu zákonu o zachování toku.

Definice 2.1.1. ***Tok*** *je funkce* $\mathbf{x} : E \to \mathbb{R}$*, kde* $\mathbf{x} \in \mathbb{R}^m$.

Funkce $\mathbf{x}$ přiřadí hodnotu $\mathbf{x}(e)$ každé hraně $e \in E$, tuto hodnotu nazveme tokem na hraně e. Takto definované toky tvoří lineární prostor (lineární kombinací toků získáme opět tok). Tok je často uvažován jako nezáporný z důvodu kompatibility s fyzikálním tokem. Uvažujme dále hlavně nezáporné toky $\mathbf{x} : E \to \mathbb{R}^+$. S tokem $\mathbf{x}$ souvisí i následující funkce $\mathbf{div_x}$ zvaná divergence.

Definice 2.1.2. *Funkci* $\mathbf{div_x} : V \to \mathbb{R}$ *nazýváme* ***divergencí*** *a definujeme ji předpisem:*

$$\mathbf{div_x}(v) = \sum_{e \in \delta^{OUT}(v)} \mathbf{x}(e) - \sum_{e \in \delta^{IN}(v)} \mathbf{x}(e).$$

Tato funkce značí rozdíl mezi hodnotou vtékající do vrcholu v a vytékající z vrcholu v pro $v \in V$. Z definice divergence plyne $\sum_{v \in V} \mathbf{div_x}(v) = 0$ (viz [1]). Tento výraz vyplývá z vlastnosti orientované hrany $e \in E$, která je jednou zahrnuta v množině vstupních hran (hodnota toku $\mathbf{x}(e)$ je v sumě přičtena) a jednou je zahrnuta v množině výstupních hran (hodnota $\mathbf{x}(e)$ je v sumě odečtena).

Dále zavedeme podobnou funkci vrcholu, kterou nazveme intenzitou vrcholu.

Definice 2.1.3. ***Intenzita vrcholu*** *je funkce* $\mathbf{b} : V \to \mathbb{R}$ *splňující* $\sum_{v \in V} \mathbf{b}(v) = 0$. *Tok* $\mathbf{x}$ *s* $\mathbf{div_x} = \mathbf{b}$*, proto splňuje*

$$\sum_{e \in \delta^{OUT}(v)} \mathbf{x}(e) - \sum_{e \in \delta^{IN}(v)} \mathbf{x}(e) = \mathbf{b}(v), \ (v \in V) \tag{2.1}$$

Výraz (2.1) lze upravit do tvaru:

$$\sum_{e \in \delta^{IN}(v)} \mathbf{x}(e) + \mathbf{b}(v) = \sum_{e \in \delta^{OUT}(v)} \mathbf{x}(e), \ (v \in V),$$

ze kterého lze vidět, že tok vtékající do vrcholu v v součtu s intenzitou vrcholu v musí dát tok vytékající z vrcholu v.

Označme funkci O jako funkci s konstantní hodnotou 0 a vektor $\mathbf{O}$ jako nulový vektor.

Definice 2.1.4. ***Cirkulace*** *v* D *je tok* $\mathbf{x}$ *s divergencí* $\mathbf{div_x} = \mathbf{O}$.

Cirkulace popisuje vlastnost zachování toku, kde tok vtékající do vrcholu v je roven toku odtékajícího z vrcholu v.

Často se uplatňují dodatečné požadavky na tok, například minimální a maximální tok jednotlivými hranami. Nechť $\mathbf{l}, \mathbf{u} : E \to \mathbb{R}$ jsou funkce takové, pro něž platí $\mathbf{l} \leq \mathbf{u}$. Porovnání vektorů je zde prováděno po složkách, tzn. $\mathbf{l}(e) \leq \mathbf{u}(e)$. Tyto funkce omezují tok $\mathbf{x}$ zdola a shora a platí $\mathbf{l} \leq \mathbf{x} \leq \mathbf{u}$. Porovnávání je prováděno opět po složkách, tzn. $\mathbf{l}(e) \leq \mathbf{x}(e) \leq \mathbf{u}(e)$. Funkci $\mathbf{l}$ nazveme dolním omezením toku a funkci $\mathbf{u}$ horním omezením toku. Pokud vyžadujeme, aby tok byl nezáporný, musí platit $\mathbf{O} \leq \mathbf{l}$. V tomto textu budeme nadále používat dolní omezení rovno nule, tj. $\mathbf{O} = \mathbf{l}$. Poté kapacita hrany $\mathbf{cap}(e) = \mathbf{u}(e) - \mathbf{l}(e) = \mathbf{u}(e) - \mathbf{O}(e) = \mathbf{u}(e)$ nabývá stejné hodnoty jako horní omezení toku $\mathbf{u}$.

Tudíž pro nezáporné toky lze psát:

$$\mathbf{O} \leq \mathbf{x} \leq \mathbf{cap}$$

nebo po složkách $0 \leq \mathbf{x}(e) \leq \mathbf{cap}(e)$, kde $e \in E$ je hranou grafu D.

Následuje Hofmannova věta [5] uvádějící charakterizaci existence cirkulace.

Věta 2.1.1 (Hoffmanova věta o cirkulaci)**.** *Nechť* $\mathbf{l}, \ \mathbf{u} : E \to \mathbb{R}$ *jsou funkce splňující* $\mathbf{l} \leq \mathbf{u}$*, které každé hraně přiřadí číslo. Potom existuje cirkulace* $\mathbf{x}$ *v orientovaném grafu* D *taková, že* $\mathbf{l} \leq \mathbf{x} \leq \mathbf{u}$ *právě tehdy, když*

$$\sum_{e \in \delta^{IN}(S)} \mathbf{l}(e) \leq \sum_{e \in \delta^{OUT}(S)} \mathbf{u}(e) \qquad (S \subseteq V).$$

Nutnou a postačující podmínku pro existenci toku nám specifikuje následující věta.

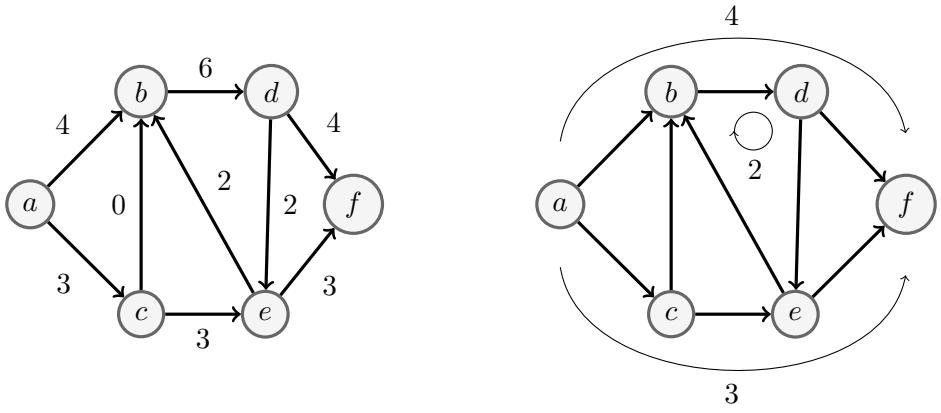
Věta 2.1.2 (Věta o existenci toku)**.** *Mějme orientovaný graf* $D = (V, E)$*. Nechť* $\mathbf{b} : V \to \mathbb{R}$ *je funkce intenzit vrcholů a* $\mathbf{cap} : E \to \mathbb{R}^+$ *funkce kapacity hran. Poté existuje tok* $\mathbf{x}$ *s intenzitou* $\mathbf{b}$*, splňující* $\mathbf{O} \leq \mathbf{x} \leq \mathbf{cap}$ *právě tehdy, když*

$$\sum_{v \in V} \mathbf{b}(v) = 0, \quad a$$

$$\sum_{v \in S} \mathbf{b}(v) \leq \sum_{e \in \delta^{OUT}(S)} \mathbf{cap}(e) \qquad (S \subseteq V)$$

2.2 Rozklady toků

Na tok se můžeme dívat jako na tok podél hran $e \in E$, nebo jako na tok podél cest a cyklů [6]. Oba tyto přístupy mají své využití. V této práci budeme většinou pracovat s tokem na hranách, viz obrázek 2.2 (vlevo). Tok podél cest a cyklů, viz obrázek 2.2 (vpravo), bude v určitém pohledu možné potkat v algoritmech řešících úlohy optimality toku. V této části textu se budeme věnovat vztahu mezi těmito dvěma alternativními přístupy a převodu mezi nimi.



Obrázek 2.2: Vyjádření toku po hranách (vlevo) a toku podél cest a cyklů (vpravo)

Pro přístup podél cest a cyklů budeme uvažovat výčet všech cest P a cyklů C v grafu D. Označme množinu všech cest symbolem $\mathcal{P}$ a množinu všech cyklů symbolem $\mathcal{C}$ v grafu D. Definujme nyní formálně tok po cestách a cyklech.

Definice 2.2.1. *Mějme funkci* $\mathbf{f} : \mathcal{P} \to \mathbb{R}$ *definovanou na množině cest* $\mathcal{P}$ *a funkci* $\mathbf{g} : \mathcal{C} \to \mathbb{R}$ *definovanou na množině cyklů* $\mathcal{C}$.

Tok na cestě P *je* $\mathbf{f}(\mathbf{P})$ *a na cyklu* $\mathbf{f}(\mathbf{C})$, *kde* $\mathbf{P} \in \mathcal{P}$ *a* $\mathbf{C} \in \mathcal{C}$.

Tok x_{uv} na hraně $(u, v) \in E$ je roven součtu toků $\mathbf{f}(P)$ a $\mathbf{g}(C)$ přes všechny cesty a cykly, které obsahují tuto hranu. Jinými slovy:

$$x_{uv} = \sum_{P \in \mathcal{P}} \delta_{uv}(P)\mathbf{f}(P) + \sum_{C \in \mathcal{C}} \delta_{uv}(C)\mathbf{g}(C),$$

kde $\delta_{uv}(P) = 1$, pokud je hrana x_{uv} obsažena v cestě P, jinak $\delta_{uv}(P) = 0$, obdobně pro $\delta_{uv}(C)$.

Následuje věta popisující převod toku po hranách na tok po cestách a cyklech a obráceně.

Věta 2.2.1 (Věta o dekompozici toku)**.** *Každý tok přes cesty* P *a cykly* C *lze jednoznačně reprezentovat jako nezáporný tok* $\mathbf{x}$ *po hranách* $e \in E$. *Naopak každý nezáporný tok* $\mathbf{x}$ *na hranách* $e \in E$ *je reprezentovatelný tokem* $\mathbf{f}$ *po cestách* P *a tokem* $\mathbf{g}$ *po cyklech* C *s dvěma následujícími vlastnostmi:*

a) Každá cesta P s kladným tokem spojuje přebytkový uzel $u \in V$ s $b(u) > 0$ s deficitním uzlem $v \in V$ s $b(v) < 0$.

b) Maximálně $n + m$ cest a cyklů má nenulový tok, z toho maximálně m cyklů má nenulový tok.

Poznamenejme, že převod toku po hranách na tok po cestách a cyklech nemusí být nutně jednoznačný.

2.3 Síť a toky v síti

V této části zavedeme základní pojmy a definice [7], které budeme dále v práci hojně využívat. Než definujeme síť a s-t síť, tak formálně představme výraz souvislosti, který je využit v pozadí těchto pojmů.

Definice 2.3.1. ***Silně souvislým grafem*** *nazveme orientovaný graf D, pokud mezi všemi dvojicemi vrcholů $u, v \in V$ existuje cesta.*

Jelikož silná souvislost grafu klade velmi přísné požadavky na orientovaný graf D, tak se v tomto textu omezíme na grafy se slabou souvislostí.

Definice 2.3.2. *Orientovaný graf D nazveme* ***slabě souvislý****, jestliže po přidání opačně orientovaných hran existuje cesta mezi všemi dvojicemi vrcholů $u, v \in V$.*

Definice 2.3.3. ***Sítí*** *N budeme nazývat orientovaný graf D.*

Pro jednoduchost budeme uvažovat síť N s konečným počtem vrcholů. Dále definujme síť se speciálními vrcholy zvanými zdroj a stok.

Definice 2.3.4. *Mějme síť N, která obsahuje vrchol zvaný zdroj $s \in V$, s nenulovým výstupním stupněm, a vrchol zvaný stok $t \in V \backslash \{s\}$, s nenulovým vstupním stupněm. Pak tuto síť nazýváme sítí s jedním zdrojem a jedním stokem. Tuto síť budeme dále značit jako* ***s-t síť****.*

Definice 2.3.5. *Síť N, na které je definovaná funkce $\mathbf{cap} : E \to \mathbb{R}$ splňující: $0 \leq \mathbf{cap}(e),\ \forall e \in E$ nazveme* ***sítí s kapacitami****.*

Pro tok $\mathbf{x}$ v síti N s kapacitami $\mathbf{cap}$ platí: $0 \leq \mathbf{x}(e) \leq \mathbf{cap}(e)$ pro všechny hrany $e \in E$.

Zaveďme pojem přípustného toku $\mathbf{x}$, který splňuje dříve uvedená omezení. Přípustný tok je klíčovým termínem v úlohách optimalizace toku, zejména při hledání maximálního toku.

Definice 2.3.6. *Nechť N je s-t síť s kapacitami.* ***Přípustný tok*** *$\mathbf{x}$ v N je funkce $\mathbf{x} : E \to R_+$, která přiřadí nezáporné reálné číslo $\mathbf{x}(e)$ hraně e, pro které platí:*

1) $0 \leq \mathbf{x}(e) \leq \mathbf{cap}(e),\ \forall e \in E$ v síti N

2) $\sum_{e \in \delta^{IN}(v)} \mathbf{x}(e) = \sum_{e \in \delta^{OUT}(v)} \mathbf{x}(e),\ \forall v \in V - \{s, t\}$ v síti N.

Definice 2.3.7. ***Velikost*** *$val(\mathbf{x})$* ***toku*** *$\mathbf{x}$ v s-t síti je tok odcházející ze zdroje s.*

$$val(\mathbf{x}) = \sum_{e \in \delta^{OUT}(s)} \mathbf{x}(e) - \sum_{e \in \delta^{IN}(s)} \mathbf{x}(e)$$

Definice 2.3.8. ***Maximální tok*** $\mathbf{x}^*$ *v síti* N *je takový tok* $\mathbf{x}$*, který má nejvyšší hodnotu* $val(\mathbf{x})$ *ze všech možných toků v síti* N*.*

S ohledem na tuto definici můžeme poznamenat, že pro maximální tok $\mathbf{x}^*$ a ostatní toky $\mathbf{x}$ v síti N platí:

$$val(\mathbf{x}) \leq val(\mathbf{x}^*)$$

Existence maximálního toku v síti N je zaručena díky Weierstrassově větě 6.0.1 a konečné síti N s omezením 1) z definice 2.3.6. Hledáním maximálního toku v síti se budeme zabývat v kapitole 4.

2.4 Řezy

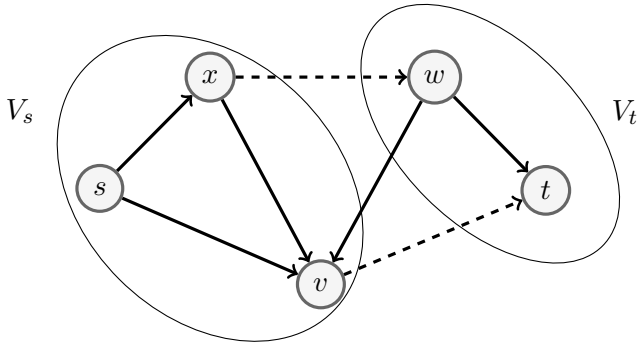
Následující část textu se bude věnovat řezům v síti. Řez v s-t síti si lze představit jako odebrání množiny hran tak, aby neexistovala cesta ze zdroje do stoku. Kapacitu řezu pak zjistíme sečtením kapacit odebraných hran.

Definice 2.4.1. *Mějme s-t síť* N*, ve které existují dvě množiny vrcholů* V_s *a* V_t*, pro které platí:*

- $V_s \subset V$ *a* $V_t \subset V$
- $s \in V_s$ *a* $t \in V_t$
- $V_s \cup V_t = V$ *a* $V_s \cap V_t = \emptyset$

Množinu všech hran e *vedoucích z množiny* V_s *do množiny* V_t *nazveme* ***s-t řez*** *sítě* N *a označíme ho* $\langle V_s, V_t \rangle$*.*

V této práci se zaměříme pouze na termín s-t řezu, který dále budeme označovat jako řez.



Obrázek 2.3: Ukázka řezu $\langle V_s, V_t \rangle$ v síti N

Na obrázku 2.3 vidíme příklad množin $V_s = \{s, x, v\}$ a $V_t = \{w, t\}$, které splňují všechny vlastnosti uvedené v definici 2.4.1. Množina hran tvořící řez $\langle V_s, V_t \rangle$ je tedy $\{(x, w), (v, t)\}$. Kdybychom pak chtěli řez $\langle V_t, V_s \rangle$, tak by mu náležela jediná hrana (w, v).

Vztah mezi toky a řezy lze vyjádřit pomocí následujícího lemmatu, které platí pro libovolné množiny V_s a V_t.

Lemma 2.4.1 (Vztah mezi toky a řezy)**.** *Mějme řez $\langle V_s, V_t\rangle$ v s-t síti N. Velikost $val(\mathbf{x})$ toku $\mathbf{x}$ v N je rovna:*

$$val(\mathbf{x}) = \sum_{e\in\langle V_s,V_t\rangle} \mathbf{x}(e) - \sum_{e\in\langle V_t,V_s\rangle} \mathbf{x}(e)$$

Definice 2.4.2. ***Kapacita řezu*** *$\langle V_s, V_t\rangle$, značená* $\mathbf{cap}\langle V_s, V_t\rangle$*, je součet kapacit hran v řezu $\langle V_s, V_t\rangle$.*

$$\mathbf{cap}\langle V_s, V_t\rangle = \sum_{e\in\langle V_s,V_t\rangle} \mathbf{cap}(e).$$

Definice 2.4.3. ***Minimální řez*** *sítě N je řez s minimální kapacitou.*

Jelikož uvažujeme konečnou síť N, počet řezů v síti je také konečný. Z konečné množiny řezů vybereme řez s nejmenší kapacitou, tudíž existence minimálního řezu je zaručena.

3 Lineární programování a dualita

Tato kapitola čerpá z textů [8] a [9], a unimodularita z článku [10].

Název této kapitoly může být zavádějící, protože slovo *programování* mělo v době pojmenování této oblasti význam spíše *plánování*, jedná se ale o optimalizaci. Lineární programování je, díky své efektivní řešitelnosti, užitečné jak v praxi, tak v teoretické oblasti. Obecnou podobu úlohy, která je řešitelná pomocí lineárního programování, můžeme zapsat následovně:

$$\begin{aligned} &\max\ \mathbf{c}^T\mathbf{x} \\ &z.\ p.\ \mathbf{A}\mathbf{x} \leq \mathbf{b},\ \mathbf{x} \in \mathbb{R}^n, \end{aligned} \tag{3.1}$$

kde $\mathbf{c} \in \mathbb{R}^n$, $\mathbf{A} \in \mathbb{R}^{m\times n}$ a $\mathbf{b} \in \mathbb{R}^m$. Lineární funkci $\mathbf{c}^T\mathbf{x} = c_1x_1 + ... + c_nx_n$ nazýváme účelovou funkcí a podmínkám ve tvaru lineárních rovnic a nerovnic říkáme omezení. Množinu všech vektorů $\mathbf{x}$ splňujících všechny podmínky nazýváme **přípustnou množinou**. Při řešení minimalizace funkce $\mathbf{c}^T\mathbf{x}$ lze problém přeformulovat na maximalizační a to ve tvaru $-\mathbf{c}^T\mathbf{x}$.

Cílem lineárního programování je najít optimální řešení $\mathbf{x}$ vyhovující všem omezením a maximalizující cílovou funkci. Pokud $\mathbf{x}$ splňuje omezující podmínky, tak ho nazýváme **přípustným řešením** úlohy (3.1). Naopak pokud vektor $\mathbf{x}$ nesplňuje alespoň jednu omezující podmínku, tak ho nazýváme **nepřípustným řešením** úlohy (3.1). Optimální řešení úlohy nemusí být jednoznačné. V některých případech může existovat nekonečně mnoho řešení maximalizujících cílovou funkci, nebo dokonce žádné. Příkladem úlohy bez řešení je úloha s neomezenou účelovou funkcí (na přípustné množině), v tomto případě budeme úlohu nazývat **neomezenou** úlohou.

Jednou z metod jak řešit úlohu lineárního programování je použití simplexového algoritmu. Jiným přístupem může být nalezení vhodného horního odhadu primární úlohy (3.1), což vede na formulaci duální úlohy, kde je naopak cílem najít nejnižší horní odhad (duální úloha bude více rozebraná v kapitole 6).

$$\begin{aligned} &\min\ \mathbf{b}^T\mathbf{y} \\ &z.\ p.\ \mathbf{A}^T\mathbf{y} \geq \mathbf{c},\ \mathbf{y} \in \mathbb{R}^m \end{aligned} \tag{3.2}$$

Přípustná řešení úlohy (3.2) jsou zároveň horními odhady primární úlohy (3.1). Tedy pro libovolná přípustná řešení primární úlohy (3.1) a duální úlohy (3.2) platí:

$$\mathbf{c}^T\mathbf{x} \leq \mathbf{b}^T\mathbf{y}$$

Tato nerovnice je nazývaná slabou dualitou.

Dále je uvedena věta o silné dualitě lineárního programování, která tvrdí, že za jemných předpokladů je hodnota optimálního řešení primární úlohy (3.1) rovna hodnotě optimálního řešení duální

úlohy (3.2).

Věta 3.0.1 (Věta o silné dualitě lineárního programování)**.** *Pro úlohy*

$$\text{maximalizovat } \mathbf{c}^T\mathbf{x} \text{ za podmínek } \mathbf{Ax} \leq \mathbf{b} \text{ a } \mathbf{x} \geq \mathbf{0} \tag{P}$$

a

$$\text{minimalizovat } \mathbf{b}^T\mathbf{y} \text{ za podmínek } \mathbf{A}^T\mathbf{y} \geq \mathbf{c} \text{ a } \mathbf{y} \geq \mathbf{0} \tag{D}$$

nastane právě jedna s následujících možností:

1. *Ani* (P)*, ani* (D) *nemá přípustné řešení.*
2. (P) *je neomezená a* (D) *nemá přípustné řešení.*
3. (P) *nemá přípustné řešení a* (D) *je neomezená.*
4. *Jak* (P)*, tak* (D) *mají přípustné řešení. Pak existuje optimální řešení* $\mathbf{x}^*$ *úlohy* (P) *a optimální řešení* $\mathbf{y}^*$ *úlohy* (D) *a platí*

$$\mathbf{c}^T\mathbf{x}^* = \mathbf{b}^T\mathbf{y}^*.$$

Tedy maximum úlohy (P) *je rovno minimu úlohy* (D)*.*

Silnou dualitou rozumíme 4. bod věty o silné dualitě.
Jednoduchý postup sestavení duální úlohy nalezneme v tabulce 3.1

	Primární úloha	Duální úloha
proměnné	$x_1, x_2, ..., x_n$	$y_1, y_2, ..., y_m$
matice	$\mathbf{A}$	$\mathbf{A}^T$
pravé strany	$\mathbf{b}$	$\mathbf{c}$
účelová funkce	max $\mathbf{c}^T\mathbf{x}$	min $\mathbf{b}^T\mathbf{y}$
	$x_j \geq 0$	$\geq$
j-té omezení má tvar	$x_j \leq 0$	$\leq$
	$x_j \in \mathbb{R}$	$=$
	$\geq$	$y_i \leq 0$
i-té omezení má tvar	$\leq$	$y_i \geq 0$
	$=$	$y_i \in \mathbb{R}$

Tabulka 3.1: Návod na sestavení duální úlohy

Kde proměnné x_j primární úlohy (P) formují omezení duální úlohy (D) a omezení primární úlohy (P) formují proměnné y_i duální úlohy (D).

Uveďme nyní důležitou vlastnost matice pro celočíselné programování.

Definice 3.0.1. *Matici* $\mathbf{A}$ *nazveme totálně unimodulární, jestliže každá její čtvercová podmatice má determinant roven* $-1, 0, 1$.

Poznamenejme, že každá matice incidence $\mathbf{A}$ orientovaného grafu D je totálně unimodulární.

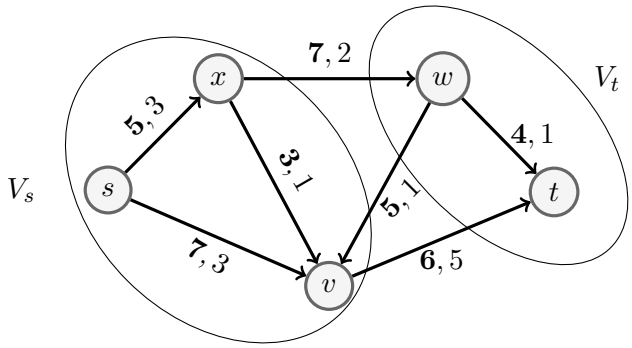
Lemma 3.0.1. *Máme-li totálně unimodulární matici* $\mathbf{A} \in \mathbb{Z}^{m \times n}$ *a vektor* $\mathbf{b} \in \mathbb{Z}^m$, *pak úloha* (P) *má celočíselné řešení* $\mathbf{x} \in \mathbb{Z}^n$.

Tato vlastnost totálně unimodulárních matic $\mathbf{A}$ se využívá například v úloze hledání maximílního toku nebo toku s minimální cenou.

4 Úloha maximálního toku

Tento text je zpracován podle zdrojů [7] a [11]. Úloha maximálního toku je jedním z problémů v teorii sítí. Hledá se zde maximální velikost toku mezi dvěma vrcholy grafu, která může být poslána skrze síť tak, aby splňovala omezující podmínky na hranách v podobě kapacit. Tento model může popisovat řadu situací z reálného světa, např. ve vodovodních sítích a kanalizacích nebo v elektrických a transportních sítích. V úloze maximálního toku máme danou s-t síť s kapacitami, což je síť se zdrojem s, stokem t a kapacitami, shora omezujícími tok procházející hranami. Cílem této problematiky je najít maximální hodnotu toku $val(\mathbf{x})$, kterou je možné poslat ze zdroje s do stoku t tak, aby tok žádnou hranou nepřekročil kapacitu dané hrany ($\mathbf{x}(e) \leq \mathbf{cap}(e)$).

Úloha maximálního toku velmi blízce souvisí s úlohou hledání minimálního řezu v s-t síti. V této úloze se hledá minimální kapacita řezu mezi komponentou V_s obsahující zdroj s a komponentou V_t obsahující stok t. Zjištění minimální kapacity řezu nám může odhalit slabá místa sítě. Například pokud budeme chtít zvýšit maximální tok sítí $val(\mathbf{x})$, tak je vhodné zvýšit kapacitu hran, které se nacházejí v minimálním kapacitním řezu $cap\langle V_s, V_t\rangle$.



Obrázek 4.1: S-t síť s velikostí toku $val(\mathbf{x}) = 6$ a kapacitou řezu $cap\langle V_s, V_t\rangle = 13$, na hranách jsou uvedené parametry: **kapacita hrany**, tok hranou

Uveďme si lemma, které nám tyto dva problémy spojuje.

Lemma 4.0.1 (Horní hranice toku)**.** *Nechť* $\mathbf{x}$ *je libovolný tok v s-t síti* N *a nechť* $\langle V_s, V_t\rangle$ *je libovolný s-t řez. Potom platí následující nerovnost.*

$$val(\mathbf{x}) \leq cap\langle V_s, V_t\rangle$$

Toto lemma popisuje horní hranici toku, která vzniká přirozeně omezením hran kapacitami. Tok hranou $\mathbf{x}(e)$ nepřekročí kapacitu hrany $\mathbf{cap}(e)$, proto také tok z komponenty V_s do komponenty V_t bude zhora omezen součtem kapacit hran vedoucích z V_s do V_t. Jelikož v lemmatu 4.0.1 uvažujeme

libovolný tok $\mathbf{x}(e)$ a libovolný řez s-t sítí, pak toto lemma platí rovněž pro maximální tok a řez s minimální kapacitou. Uveďme si zde slabou dualitu toku, která vyplývá z lemmatu 4.0.1.

Lemma 4.0.2 (Slabá dualita)**.** *Mějme s-t síť N. Nechť $\mathbf{x}^*$ je maximální tok v N a nechť $\langle V_s^*, V_t^* \rangle$ je minimální řez v N. Potom*

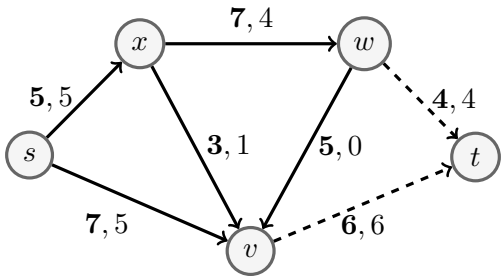
$$val(\mathbf{x}^*) \leq cap\langle V_s^*, V_t^* \rangle$$

V následující větě je uvedena věta o silné dualitě, která je speciálním případem silné duality lineárního programování.

Věta 4.0.1 (Ford, Fulkerson [12])**.** *Hodnota maximálního toku v síti N je rovna kapacitě minimálního řezu.*

$$val(\mathbf{x}^*) = cap\langle V_s^*, V_t^* \rangle$$

Tok v s-t síti na obrázku 4.2 má po maximalizaci hodnotu 10 a řez $cap\langle\{s, x, v, w\}, \{t\}\rangle = 10$.



Obrázek 4.2: Maximální tok a minimální řez

4.1 Formulace úlohy pomocí LP

Úloha maximálního toku je řešitelná pomocí lineárního programování. Následující zápis poskytuje formulaci primární úlohy, tedy úlohy hledání maximálního toku sítí N. Pro tento zápis uvažujme s-t síť N, tok sítí $\mathbf{x} : E \to \mathbb{R}$ a kapacity hran $\mathbf{cap} : E \to \mathbb{R}^+$.

$$\max \sum_{e \in \delta^{OUT}(s)} \mathbf{x}(e) \; - \sum_{e \in \delta^{IN}(s)} \mathbf{x}(e)$$

$$z.p. \sum_{e \in \delta^{IN}(v)} \mathbf{x}(e) = \sum_{e \in \delta^{OUT}(v)} \mathbf{x}(e) \qquad \forall v \in V \backslash \{s, t\}$$

$$\mathbf{x}(e) \leq \mathbf{cap}(e) \qquad \forall e \in E$$

$$0 \leq \mathbf{x}(e) \qquad \forall e \in E$$

Formulace duální úlohy [13] je sestavena podle tabulky 3.1 a souvisí s problematikou hledání minimálního řezu $\langle V_s, V_t \rangle$ v s-t síti. Duální proměnné z se vztahují k hranám sítě a proměnné y k vrcholům.

$$\min \sum_{e \in E} \mathbf{cap}(e) z_e$$

$$z.p.\ y_u - y_v + z_e \geq 0 \qquad \forall e = (u,v) \in E$$

$$-y_s + y_t \geq 1 \qquad (pro\ \mathbf{x}^*)$$

$$y_v \in \mathbb{R} \qquad \forall v \in V$$

$$z_e \geq 0 \qquad \forall e \in E$$

4.2 Algoritmy

V této části uvedeme dva vybrané algoritmy, kterými se dá řešit úloha maximálního toku v s-t síti.

4.2.1 Fordův Fulkersonův algoritmus

Jednou z myšlenek při řešení úlohy maximálního toku a minimálního řezu je hledání vhodné rozšiřující cesty v s-t síti, podél níž je možné navýšit aktuální velikost toku. S touto strategií pracuje jeden z prvních algoritmů na hledání maximálního toku Ford-Fulkersonův algoritmus. Následně byl tento přístup vylepšen v Edmons-Karpově algoritmu nalezením nejkratší rozšiřující cesty. Dinicův algoritmus také pracuje na podobném přístupu, ovšem hledá najednou všechny nejkratší rozšiřující cesty (více viz [14]).

Definice 4.2.1. ***Kvazi-cestou*** *v s-t síti nazveme cestu Q s libovolně orientovanými hranami, která má počátek ve vrcholu s a konec ve vrcholu t.*

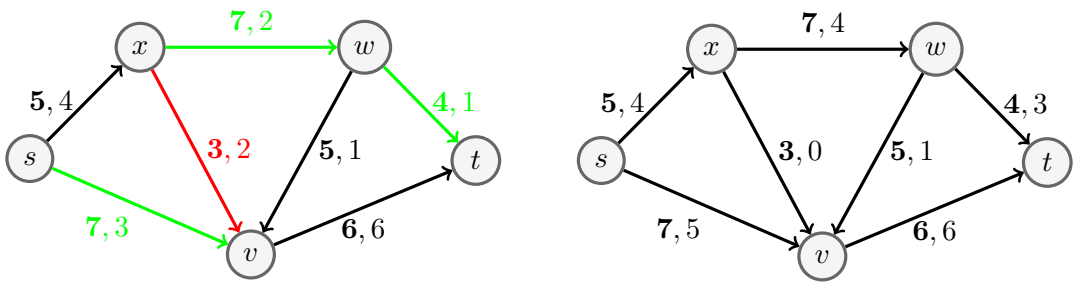
$$Q = (s = v_1, ..., v_i, v_{i+1}, ..., v_k = t)$$

Pokud je hrana e_i na cestě Q orientovaná z vrcholu v_i do vrcholu v_{i+1}, tak tuto hranu nazveme ***orientovanou dopředu****, naopak hranu vedoucí z vrcholu v_{i+1} do vrcholu v_i nazveme hranou* ***orientovanou dozadu****.*

Definice 4.2.2. ***Rozšiřující cestou*** *toku $\mathbf{x}$ nazveme kvazi-cestu Q, jejíž tok můžeme navýšit na hranách orientovaných dopředu a snížit na hranách orientovaných dozadu (o nenulové hodnoty).*

Příklad rozšiřující cesty můžeme vidět na obr. 4.3, zde jsou zelené hrany orientované dopředu a červená hrana orientovaná dozadu. Změnu toku na rozšiřující cestě získáme jako minimum z $\mathbf{cap}(e) - \mathbf{x}(e)$ pro hrany orientované dopředu a $\mathbf{x}(e)$ pro hrany orientované dozadu. Tuto hodnotu poté přičteme k toku na každé hraně orientované dopředu a odečteme od toku na každé hraně orientované dozadu.

Věta 4.2.1 (Charakteristika maximálního toku, [15])**.** *Mějme tok $\mathbf{x}$ v s-t síti N. Potom $\mathbf{x}$ je maximální tok v N právě tehdy, když neexistuje vylepšující cesta v N.*



Obrázek 4.3: Rozšiřující cesta v s-t síti

4.2.2 Algoritmus Push a Relabel

Jinou úvahou hledání maximálního toku, která je použita např. v Goldbergově algoritmu, je metoda Push and Relabel [16]. Tento přístup využívá ideu přebytku toku neboli přetoku vrcholu $v \in V$ v síti bez orientace hran. Přetok $p(v) = \sum_{u \in V} \mathbf{x}(u, v) - \sum_{u \in V} \mathbf{x}(v, u)$ je rozdíl toku vtékajícího do vrcholu v a toku vytékajícího z vrcholu v. Uvažujme dále tok $\mathbf{x}$ s přetokem p takovým, že tok $\mathbf{x}$ splňuje omezení na kapacity, ale množství toku vtékající do vrcholu je větší nebo rovno množství toku vytékající z vrcholu. Poté je přetok vrcholu vždy $p(v) \geq 0$. Dále budeme také používat pojem reziduální kapacity hrany $\mathbf{r_f}(v, w) = \mathbf{cap}(v, w) - \mathbf{x}(v, w)$, která nám udává množství toku, které můžeme přidat na hraně e, aby nebyla překročena kapacita hrany $\mathbf{cap}(e)$.

Definujme základní operace, které tento algoritmus využívá:

- Operace *PUSH* (v, w) se smí provést, pokud je vrchol v aktivní ($p(v) > 0$), reziduální kapacita je kladná ($r_f(v, w) > 0$) a pro vzdálenosti vrcholů v, w od stoku t platí $d(v) = d(w) + 1$. Poté se určí hodnota $\delta = \min\{p(v), r_f(v, w)\}$ a upraví se následující hodnoty $\mathbf{x}(v, w) + \delta$, $p(v) - \delta$ a $p(w) + \delta$.

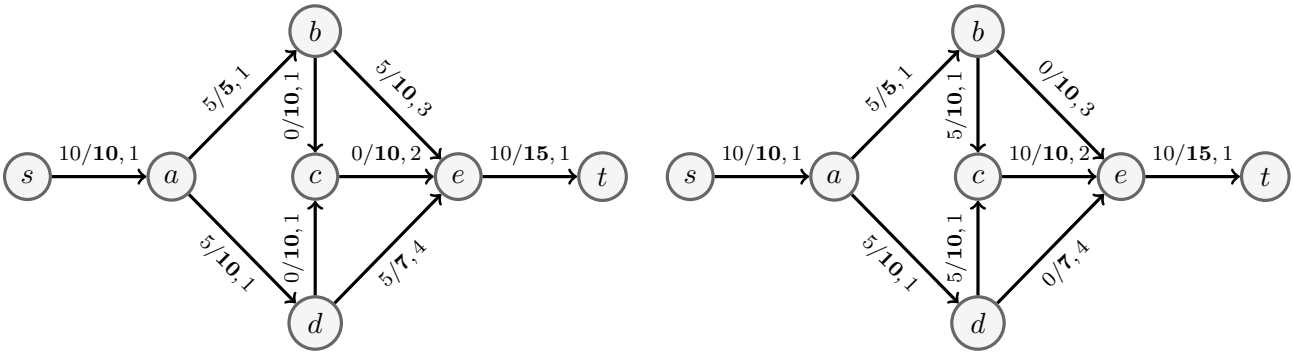
- Operace *RELABEL* v se smí provést, pokud je vrchol v aktivní ($p(v) > 0$) a všechny sousední vrcholy $w \in V$, pro které platí $r_f(v, w) > 0$, mají větší nebo stejnou vzdálenost ($d(v) \leq d(w)$) od stoku t. Poté se vypočítá nová vzdálenost vrcholu v od stoku pomocí tvaru $d(v) = \min\{d(w) + 1 \mid (v, w) \in E\}$.

Nejprve se na hranách vedoucích ze zdroje $e = (s, v)$, $v \in V$ nastaví přetok $\mathbf{x}(e) = \mathbf{cap}(e)$ a pro zbylé hrany $\mathbf{x}(e) = 0$. Dále vypočteme přebytek každého vrcholu $p(v) = \sum_{e:e=(u,v)} \mathbf{x}(e) - \sum_{e:e=(v,u)} \mathbf{x}(e)$ mimo zdroj a stok, a nastavíme vzdálenost $\mathbf{d}$ vrcholu v od stoku t: $\mathbf{d}(s) = n$ a $\mathbf{d}(v) = 0$, $\forall v \in V - \{s\}$. Poté se opakovaně v libovolném pořadí provádí operace *PUSH* a *RELABEL* dokud existují aktivní vrcholy. Pokud neexistují žádné aktivní vrcholy, tak algoritmus končí a přetok je označen za maximální tok v s-t síti.

5 Toky s minimální cenou

Pro zpracování tohoto textu byl použit zdroj [6], [15].

V předchozí kapitole jsme se věnovali hledání maximálního toku v s-t síti. Další úlohou, kterou se budeme zabývat bude hledání b-toku s minimální cenou. Představme si ji na úloze hledání maximálního toku s minimální cenou v s-t síti.



Obrázek 5.1: Maximální tok s cenou toku 65 (vlevo), maximální tok s cenou 60 (vpravo), na hranách jsou uvedené parametry: tok hranou/ **kapacita hrany**, cena na hraně

Na obrázku 5.1 máme zobrazené dvě varianty maximálního toku v s-t síti. Cena toku, kterou vypočteme vzorečkem $\mathbf{c}(\mathbf{x}) = \sum_{e \in E(N)} \mathbf{x}(e)\mathbf{c}(e)$, má vlevo hodnotu 65 a vpravo hodnotu 60 i přesto, že se jedná o stejnou velikost toku $val(\mathbf{x}) = 10$. Volbou vhodného toku tedy můžeme značně ušetřit. V této úloze navíc budeme uvažovat síť, která může obsahovat více zdrojů a více stoků. Poté se nejedná o problematiku hledání maximálního toku s minimální cenou, ale o úlohu hledání *b-toku* s minimální cenou.

Definice 5.0.1. *Mějme síť N s kapacitami hran $\mathbf{u} : E(N) \to \mathbb{R}_0$ a intenzitami vrcholů $\mathbf{b} : V(N) \to \mathbb{R}$, pro které platí $\sum_{v \in V(N)} \mathbf{b}(v) = 0$. **B-tok** je funkce $\mathbf{x} : E(N) \to \mathbb{R}^+$ splňující $0 \leq \mathbf{x}(e) \leq \mathbf{u}(e)$ pro všechny hrany $e \in E(N)$ a $\sum_{e \in \delta^{OUT}(v)} \mathbf{x}(e) - \sum_{e \in \delta^{IN}(v)} \mathbf{x}(e) = \mathbf{b}(v)$pro všechny vrcholy $v \in V(N)$.*

Ve vrcholech se zápornou intenzitou $b < 0$ se tok spotřebovává, proto takové vrcholy můžeme nazvat stoky. Kdežto vrcholy s kladnou intenzitou $b > 0$ tok vytvářejí a proto je můžeme nazývat zdroje.

Připomeňme větu o existenci toku 2.1.2, která platí i pro b-tok.

Věta 5.0.1. *Mějme síť s kapacitami N a funkci intenzit $\mathbf{b}(v) : V(N) \to \mathbb{R}$, pro kterou platí $\sum_{v \in V(N)} \mathbf{b}(v) = 0$. Poté existuje b-tok právě tehdy, když platí*

$$\sum_{e \in \delta^{OUT}(X)} \mathbf{u}(e) \geq \sum_{v \in X} \mathbf{b}(v) \qquad \forall X \subseteq V(N)$$

Věnujme se nyní kritériu optimality. Uvažujme symetrizovanou síť N_x, kde ke každé existující hraně $e = (u, v)$ v N přidáme opačně orientovanou hranu $e = (v, u)$, které přiřadíme cenu $c(v, u) = -c(u, v)$. V takto definované reziduální síti N_x je cenová funkce $\mathbf{c}$ nezávislá na změně b-toku.

Definice 5.0.2. *Mějme síť s kapacitami N a b-tok $\mathbf{x}$. **Rozšiřující cyklus** je cyklus v reziduální síti N_x, podél kterého můžeme měnit cenu b-toku.*

Věta 5.0.2 (Klein (1967), [15])**.** *B-tok $\mathbf{x}$ v síti $(N, \mathbf{u}, \mathbf{b}, \mathbf{c})$ má minimální cenu právě tehdy, když neexistuje rozšiřující cyklus se zápornou cenou.*

5.1 Formulace úlohy pomocí LP

Problém toků s minimální cenou zapsaný pomocí lineárního programování:

$$\min \sum_{e \in E(N)} \mathbf{c}(e) x_e$$

$$z.p. \sum_{e \in \delta^{OUT}(v)} x_e - \sum_{e \in \delta^{IN}(v)} x_e = \mathbf{b}(v) \quad (v \in V(N))$$

$$x_e \leq u_e \quad (e \in E(N))$$

$$x_e \geq 0 \quad (e \in E(N))$$

a duální úlohy:

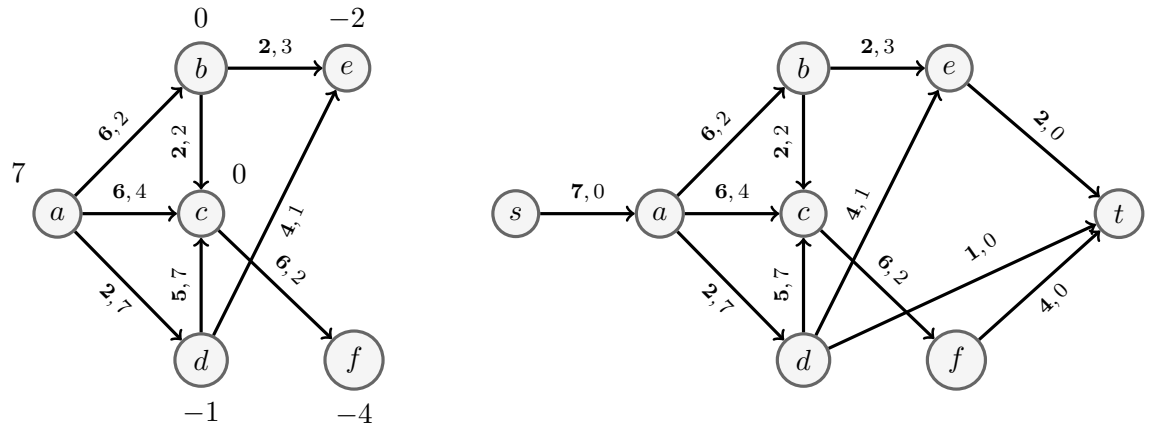
$$\max \sum_{v \in V(N)} -b_v y_v + \sum_{e \in E(N)} -u_e z_e$$

$$z.p. y_v - y_w + z_e \geq -\mathbf{c}(e) \quad (e = (v, w) \in E(N))$$

$$z_e \geq 0 \quad (e \in E(N))$$

5.1.1 Hledání přípustného řešení

Pokud budeme chtít řešit úlohu toku s minimální cenou pro síť s více než jedním zdrojem a jedním stokem viz Obrázek 5.2 (vlevo), tak můžeme síť převést na s-t síť viz Obrázek 5.2. S-t síť vytvoříme přidáním vrcholu s reprezentující zdroj a vrcholu t reprezentující stok. Vrchol s poté spojíme se všemi vrcholy u s kladnou intenzitou $b(u) \geq 0$ a kapacitu nastavíme na $cap(s, u) = b(u),\ u \in V$. Vrchol t propojíme se všemi vrcholy u se zápornou intenzitou $b(u) \leq 0$ a kapacitu těchto hran nastavíme na $cap(u, t) = -b(u),\ u \in V$. Cena přidaných hran je nulová. V této vytvořené s-t síti hledáme maximální tok s minimální cenou. Velikost $val(\mathbf{x})$ maximálního toku $\mathbf{x}$ v nově vytvořené s-t síti je roven celkové produkci vrcholů $\mathbf{b}(v) > 0$ v původní síti N.



Obrázek 5.2: Tok s více zdroji a stoky (vlevo), tok s jedním zdrojem a jedním stokem (vpravo), na hranách jsou uvedené parametry: **kapacita hrany**, cena hrany

5.2 Algoritmy

Mezi algoritmy řešící úlohu minimálního toku patří například: algoritmus rušení záporných cyklů, síťový simplex a algoritmus škálování kapacity. Tato část byla čerpána ze zdrojů: [17], [18].

5.2.1 Residuální graf

V našich algoritmech pracujeme s residuálním grafem, kde jsou vrcholům přiřazeny duální proměnné, jež označujeme jako potenciály, a hranám přiřazujeme duální proměnné, známé jako reziduální ceny.

5.2.2 Rušení záporných cyklů

V tomto algoritmu uvažujeme zadaný b-tok, jehož cenu budeme chtít dále minimalizovat. Algoritmus se provádí ve třech hlavních krocích: definováním reziduální sítě, hledáním cyklů se zápornou cenou a úpravou toku.

Začněme definováním reziduální sítě $N_x = (V_x, E_x)$, kde vrcholy reziduální sítě jsou $V_x(N_x) = V(N)$ a mezi hrany E_x reziduální sítě N_x patří hrany E z původní sítě N, pro které platí $\mathbf{cap}(e) - \mathbf{x}(e) < 0$, reziduální cena těchto hran je $\mathbf{c}_x(e) = \mathbf{c}(e)$. Do množiny E_x přidáme dále hrany opačně orientované k hranám $e \in E$, pro které platí $\mathbf{x}(e) > 0$, které mají reziduální cenu $\mathbf{c}_x(e) = -\mathbf{c}(e)$. Tyto hrany E_x mají tok rovný reziduální kapacitě:

$$\mathbf{r}_x(e) = \begin{cases} e = (u, v) \in E : \mathbf{cap}(e) - \mathbf{x}(e) \\ e = (v, u) \notin E : \mathbf{x}(e) \end{cases}$$

V takto definované reziduální síti N_x hledáme cyklus $\mathcal{C}$, podél něhož má tok zápornou cenu. Pokud takový cyklus $\mathcal{C}$ nalezneme, tak zjistíme parametr $\delta = \min\{e \in \mathcal{C} : \mathbf{r}_x(e)\}$, který určí, jak velký tok budeme v původní síti N opravovat. Dále upravíme tok hran původní sítě N, které jsou obsaženy v

cyklu $\mathcal{C}$. Pokud je hrana $e \in E$ orientovaná souhlasně s cyklem $\mathcal{C}$, tak tok touto hranou je $\mathbf{x}(e) + \delta$. Pokud je hrana $e \in E$ orientovaná opačným směrem, tak tok touto hranou je $\mathbf{x}(e) - \delta$.

V tomto algoritmu se hodnota b-toku minimalizuje hledáním zápornách cyklů v reziduální síti a následně jejich rušením. Pokud v reziduální síti neexistuje žádný záporný cyklus, tak se algoritmus ukončí s optimálním řešením $\mathbf{x}$.

5.2.3 Síťový simplex

V tomto algoritmu je využita kostra sítě, kterou budeme značit T. Jedná se o síť, ze které jsou odebrány hrany tak, aby síť zůstala souvislá, ale neobsahovala žádné cykly v neorientované verzi. Princip síťového simplexu spočívá v hledání kostry s minimální cenou toku tak, aby výsledný tok splňoval podmínky b-toku. Tento algoritmus se skládá ze tří kroků. Nejprve nalezne přípustnou kostru sítě, následně rozhodne zda se jedná o optimální kostu či nikoli. Pokud se nejdená o optimální kostru, tak ji vylepší a opakuje tyto tři kroky do doby, než najde optimální kostru.

Nyní představíme jednotlivé kroky síťového simplexu podrobněji a začneme inicializací. Mějme tedy kostru $T = E_T \backslash E$ s hranami E_T a tokem $\mathbf{x}$ splňujícím podmínky b-toku na této kostře. Takovouto kostru T budeme dále nazývat přípustnou. Dále mějme množinu hran L, pro které platí $\mathbf{x}(e) = 0$ a množinu hran U, do které patří hrany $\mathbf{x}(e) = \mathbf{cap}(e)$. Množiny L a U tedy na začátku algoritmu nastavme $L = E$ a $U = \emptyset$. Nastavme také potenciál $\pi(x) = 0$ pro náš libovolně zvolený vrchol $x \in V$ a potenciály ostatních vrcholů dopočteme $c_e - \pi(u) + \pi(v) = 0,\ e = (u, v) \in E_T$.

V dalším kroku se poté otestuje, zda je kostra T optimální. Pokud není v $L \cup U$ hrana $e \in L : c_\pi(e) < 0$ nebo $e \in U :\ c_\pi(e) > 0$, tak se algoritmus zastaví.

Vyberme hranu v $L \cup U$ splňující $e \in L :\ c_\pi(e) < 0$ nebo $e \in U :\ c_\pi(e) > 0$ a najděme cyklus C v $T \cup e$. Dále uvažujme cyklus C orientovaný ve směru hrany e a určeme parametr

$$\delta = \min \begin{cases} \mathbf{cap}(e) - \mathbf{x}(e) \text{ pro hrany orientované ve směru } C \\ \mathbf{x}(e) \text{pro hrany orientované proti směru} C \end{cases}$$

Tok $\mathbf{x}$ tedy rozšíříme tak, že k toku na hranách orientovaných po směru C přičteme δ: $\mathbf{x}(e) + \delta$, a k tokům na hranách orientovaných proti směru C odečteme δ: $\mathbf{x}(e) - \delta$. Jedna z hran cyklu C dosáhne toku $\mathbf{x}(e) = \mathbf{cap}(e)$, nebo $\mathbf{x}(e) = 0$. Označme tuto hranu písmenem a.

V dalším kroku upravíme kostru T tak, abychom dále pracovali znovu s kostrou sítě. Do množiny T přidáme hranu e a naopak odebereme hranu a. Z množiny L odebereme hranu e a přidáme hranu a, pouze pokud tok na hraně a je $\mathbf{x}(a) = 0$. Množina U jsou poté zbylé hrany sítě, které nepatří do množin T a L. Po upravení množin T, L, U znovu vypočteme potenciál $\pi(v)$ pro všechny vrcholy $v \in V - \{x\}$ s využitím hodnoty $\pi(x) = 0$. Následuje znovu krok s testováním oprimality kostry sítě.

6 Konvexní programování

V této kapitole se budeme zabývat nelineární optimalizací úloh , konkrétně konvexním programováním [19], [20].

Optimalizační problém má obecně podobu

$$\textit{minimalizovat nebo maximalizovat } f(\mathbf{x})$$

$$\textit{za podmínek } \mathbf{x} \in S$$

$$\textit{minimalizovat nebo maximalizovat } f(\mathbf{x})$$

$$\textit{za podmínek } \mathbf{x} \in S$$

kde S je přípustná množina v $\mathbb{R}^n$ a $f(\mathbf{x})$ je reálná funkce definovaná na S. Nelineární optimalizační úloha obsahuje nelineární cílovou funkci, nebo nelineární podmínky definující množinu S. Úlohy lze mezi sebou převádět a z maximalizační úlohy se dá snadno vyrobit úloha minimalizační:

$$\max\{f(x) : x \in S\} = -\min\{-f(x) : x \in S\}.$$

Dále budeme optimalizační problém uvažovat ve formě

$$\min_{x \in S} f(x) \tag{6.1}$$

kde $S = \{x \in \mathbb{R}^n : g_i(x) \leq 0,\ i = 1, ..., p\}$.

Definice 6.0.1. *Množinu nazveme kompaktní, jestliže je uzavřená a omezená zároveň.*

Nutno poznamenat, že ne vždy existuje minimum funkce. Většinou je nalezeno infimum cílové funkce z důvodu nekompaktnosti množiny, nebo nespojitosti funkce na množině. Jelikož v úloze 6.1 množina S je uzavřená a omezená, tak následující věta zaručuje existenci řešení v naší úloze. Následující věta popisuje postačující podmínky pro existenci minima.

Věta 6.0.1 (Weierstrassova věta)**.** *Mějme neprázdnou kompaktní množinu $S \subseteq \mathbb{R}^n$ a spojitou funkci na S $f : S \to \mathbb{R}$. Potom funkce $f(x)$ má alespoň jedno globální minimum (maximum) na množině S*

6.1 Úvod do konvexního programování

Pro formulaci úlohy konvexního programování si nejprve definujme pojmy jako je konvexní množina a konvexní funkce. Pro definování pojmu konvexní množiny použijeme konvexní kombinaci bodů.

Definice 6.1.1. *Mějme body $x_1, x_2, ..., x_n \in S$, kde $S \subseteq \mathbb{R}^n$ a nezáporná čísla $\lambda_1, \lambda_2, ..., \lambda_n$, pro která platí $\sum_{i=1}^{n} \lambda_i = 1$. Bod x je **konvexní kombinací** bodů, pokud platí:*

$$x = \sum_{i=1}^{n} \lambda_i x_i.$$

V definici konvexní množiny využijeme konvexní kombinace dvou bodů.

Definice 6.1.2. *Množina $S \subseteq \mathbb{R}^n$ je **konvexní množina**, jestliže pro libovolné dva prvky $x, y \in S$ a pro libovolný parametr $\alpha \in \langle 0, 1 \rangle \in S$ platí*

$$\alpha x + (1 - \alpha) y \in S.$$

Požadujeme tedy, aby úsečka mezi libovolnými dvěma body $x, y \in S$ ležela rovněž v konvexní množině $S \subseteq \mathbb{R}^n$. Příkladem konvexní množiny může být množina na obrázku 6.1 (vlevo). Mezi konvexní množiny náleží i prázdná množina $\emptyset$ nebo prostor $\mathbb{R}^n$. Poznamenejme, že průnikem dvou konvexních množin je opět konvexní množina. Tato vlastnost je značně užitečná pro konvexní optimalizaci, protože dovoluje kombinovat konvexní podmínky v konvexní optimalizační úloze.

Obrázek 6.1: Příklad konvexní množiny (vlevo) a nekonvexní množiny (vpravo)

Na obrázku 6.1 (vpravo) můžeme také vidět příklad nekonvexní množiny. Množina vpravo není konvexní, protože konvexní kombinace bodů $x, y \in S$ není uzavřenou operací na množině S, takže některé prvky vzniklé konvexní kombinací prvků $x, y \in S$ neleží v množině S.

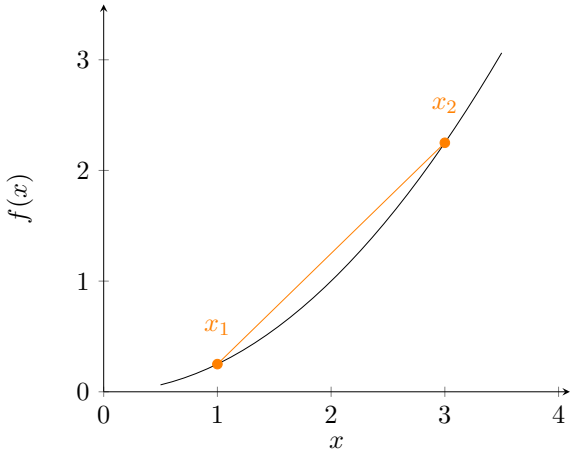
Dále definujme konvexní funkci.

Definice 6.1.3. ***Konvexní funkce** je funkce $f : S \to \mathbb{R}$ taková, že pro libovolné dva prvky $x_1, x_2 \in S \subseteq \mathbb{R}^n$ a parametr $\alpha \in \langle 0, 1 \rangle$ a platí:*

$$f(\alpha x_1 + (1 - \alpha) x_2) \leq \alpha f(x_1) + (1 - \alpha) f(x_2). \tag{6.2}$$

Na levé straně nerovnice (6.2) je funkce použitá na konvexní kombinaci bodů x_1, x_2, kdežto na pravé straně je konvexní kombinace funkčních hodnot x_1, x_2. V případě konvexní funkce bude levá strana vždy menší nebo rovna pravé straně. Rovnost těchto stran nastane v případě, když funkce bude lineární, což je krajní případ konvexní funkce. Na obrázku 6.2 můžeme vidět vizualizaci nerovnice (6.2). Černou barvou je vykreslená levá strana nerovnice a oranžovou barvou je znázorněná pravá strana.

Důležitou vlastností konvexních funkcí v oblasti konvexní optimalizace je vztah mezi lokálními a globálními minimy. Lokální minimum konvexní funkce je díky vlastnostem konvexity zároveň globálním minimem této funkce.



Obrázek 6.2: Vizualizace definice konvexní funkce

6.2 Formulace úlohy konvexního programování

Úloha konvexního programování je optimalizační úloha, jejímž cílem je nalézt optimalizační proměnnou $\mathbf{x} \in \mathbb{R}^n$ minimalizující cílovou funkci $f(\mathbf{x})$. Tato úloha je často doplněna o podmínky, které omezují množinu S, na které hledáme optimum. Formulujme si úlohu následovně:

$$\begin{array}{lll} \text{minimalizovat} & f(\mathbf{x}) & \\ \text{za podmínek:} & g_i(\mathbf{x}) \leq 0 & \forall i \in 1, 2, ..., I \\ & h_j(\mathbf{x}) = 0 & \forall j \in 1, 2, ..., J, \end{array}$$

kde cílová funkce $f(\mathbf{x})$ je konvexní funkcí. Podmínky $g_i(\mathbf{x})$ nazveme podmínkami typu nerovnosti a požadujeme, aby byly konvexními funkcemi. Kdežto od podmínek typu rovnosti $h_j(\mathbf{x})$ vyžadujeme, aby byly afinními funkcemi. Každá z podmínek g_i a h_j definují konvexní množinu, jejichž průnikem získáme opět konvexní množinu, kterou budeme značit S. Hledání globálních minim konvexní funkce $f(\mathbf{x})$ na konvexní množině S v tomto případě odpovídá úloze hledání lokálních minim funkce. Formulaci úlohy konvexního programování můžeme také zapsat pomocí následujícího výrazu:

$$\min_{\mathbf{x} \in S} f(\mathbf{x}). \tag{6.3}$$

Optimalizační proměnnou $\mathbf{x}$ splňující podmínky úlohy nazveme přípustným řešením úlohy(6.3). Optimálním řešením úlohy (6.3) nazveme takové přípustné řešení $\mathbf{x}^*$, pro které platí $f(\mathbf{x}^*) \leq f(\mathbf{x})$.

6.3 Dualita v konvexním programování

V této části textu zavedeme duální úlohu k primární úloze (6.3). Jedná se o alternativní pohled na primární úlohu, který nám může pomoci odvodit některé její vlastnosti. Budeme se zde také zabývat tím, že optimalizace primární a duální úlohy vedou k ekvivalentním výsledkům. V určitých

případech může být řešení duální úlohy výrazně jednodušší, což je další důvod, proč dualitu jako takovou zmiňujeme. Zaveďme nejprve max-min nerovnost. (Uvažujme, že všechna použitá maxima i minima existují.)

Věta 6.3.1. *Pro libovolnou funkci* $f : X \times Y \to \mathbb{R}$*:*

$$\max_{x\in X}\min_{y\in Y} f(x,y) \leq \min_{y\in Y}\max_{x\in X} f(x,y)$$

Důkaz. Pro libovolná $x' \in X$ a $y' \in Y$ platí:

$$\min_{y\in Y} f(x',y) \leq f(x',y').$$

Pokud je tedy předchozí výraz pravdivý pro libovolné $x' \in X$, pak je také pravdivý pro x s maximální hodnotou:

$$\max_{x\in X}\min_{y\in Y} f(x,y) \leq \max_{x\in X} f(x,y').$$

Tento výraz je pravdivý pro všechna $y' \in Y$, tedy je pravdivý i pro minimální y a platí:

$$\max_{x\in X}\min_{y\in Y} f(x,y) \leq \min_{y\in Y}\max_{x\in X} f(x,y)$$

□

Větu 6.3.1 můžeme slovně interpretovat následujícím způsobem. Maximum z minimálních hodnot je menší nebo rovno minimu z maximálních hodnot. Pokud nastane rovnost, tak bod (x, y) nazveme sedlovým bodem funkce f.

Uvažujme nyní primární optimalizační úlohu P ve tvaru:

$$\min f(\mathbf{x}) \quad \text{za podmínek: } g_i(\mathbf{x}) \leq 0,\ i = 1, 2, ..., I. \tag{P}$$

Zaveďme Lagrangeovu funkci nazývanou Lagrangian $\mathcal{L}$ primární úlohy s lagrangeovými multiplikátory $\lambda_i \geq 0,\ i = 1, 2, ..., I$:

$$\mathcal{L} = f(\mathbf{x}) + \sum_{i=1}^{I} \lambda_i g_i(\mathbf{x})$$

Pokud jsou všechny podmínky $g_i(\mathbf{x}) \leq 0$ splněny, tak je suma $\sum_{i=1}^{I} \lambda_i g_i(\mathbf{x})$ jistě záporná nebo nulová. Za předpokladu, že je alespoň jedna podmínka nesplněná, tedy $g_i(\mathbf{x}) > 0$, tak při zvyšování parametru $\lambda_i(\mathbf{x}) \to \infty$ bude i $\mathcal{L}(\mathbf{x}, \lambda) \to \infty$. Jelikož se ale snažíme o minimalizaci $\mathcal{L}(\mathbf{x}, \lambda)$, tak tato varianta není příliš přívětivou. Zavedení Lagrangianu nám tedy dává jiný způsob zápisu cílové funkce primární úlohy (P):

$$\mathcal{P}(\mathbf{x}) = \max_{\lambda\geq 0} \mathcal{L}(\mathbf{x}, \lambda)$$

Cílovou funkci $\mathcal{P}(\mathbf{x})$ chceme minimalizovat přes $\mathbf{x}$, tedy:

$$\min_{\mathbf{x}} \mathcal{P}(\mathbf{x}) = \min_{\mathbf{x}} \max_{\lambda \geq 0} \mathcal{L}(\mathbf{x}, \lambda)$$

Na úlohu použijeme dříve zmíněnou větu 6.3.1 a získáme tak dolní odhad primární úlohy.

$$\max_{\lambda \geq 0} \min_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \lambda) \leq \min_{\mathbf{x}} \max_{\lambda \geq 0} \mathcal{L}(\mathbf{x}, \lambda)$$

Zaveďme duální cílovou funkci $\mathcal{D}$ popisující spodní hranici primární úlohy, kterou budeme chtít maximalizovat vzhledem k λ:

$$\mathcal{D}(\lambda) = \min_{\mathbf{x}} \mathcal{L}(\mathbf{x}, \lambda).$$

Formulujme duální úlohu k primární úloze (P).

$$\max \mathcal{D}(\lambda) \quad \text{za podmínek: } \lambda_i \geq 0,\ i = 1, 2, ..., I. \tag{D}$$

Uveďme nyní větu o slabé dualitě.

Věta 6.3.2 (Slabá dualita)**.** *Nechť vektor $\mathbf{x}^*$ je globálním minimem primární úlohy. Poté pro každou $\lambda \geq 0$ platí*

$$\mathcal{D}(\lambda) \leq \mathcal{D}(\lambda^*) \leq f(\mathbf{x}^*),$$

kde λ^* je globálním maximem duální úlohy.

Rozdíl hodnot $f(\mathbf{x}^*) - \mathcal{D}(\lambda^*)$ se nazývá dualitní rozdíl, pokud je roven nule, tak platí silná dualita.

$$\max_{\lambda \geq 0} \mathcal{D}(\lambda) = \min_{\mathbf{x}} \mathcal{P}(\mathbf{x})$$

Zatímco slabá dualita platí vždy, silná dualita platí pouze za určitých podmínek. Uveďme nyní Slaterovu podmínku, která bude využita ve větě zaručující silnou dualitu.

Slaterova podmínka:
Omezení $g_i(\mathbf{x})$ jsou konvexní pro $(i = 1, ..., I)$ a existuje $\overline{\mathbf{x}}$ splňující $g_i(\overline{\mathbf{x}}) < 0$ $(i = 1, ..., I)$ $\wedge \mathbf{A}\overline{\mathbf{x}} = \mathbf{b}$.

Věta 6.3.3 (Slaterova, [21])**.** *Pokud je optimalizační problém konvexní a platí Slaterova podmínka, tak platí také silná dualita.*

6.4 Karush-Kuhn-Tuckerovy podmínky

V této části představíme nutné podmínky pro existenci lokálního minima systémem rovnic a nerovnic, které se nazývají Karush-Kuhn-Tuckerovy podmínky. Díky konvexitě cílové funkce a přípustné množiny S, tyto podmínky platí i pro globální minimum. V následující větě zavádíme nutné podmínky pro úlohu 6.3 s podmínkami typu nerovnosti g_i.

Věta 6.4.1. *Každá z následujících podmínek je kvalifikace omezení:*

1. $g_i(x) = a_i^T x - b_i$, $a_i \in \mathbb{R}^n \backslash \{0\}$, $b_i \in \mathbb{R}$ $(i = 1, ..., I)$ *(podmínka linearity)*

2. $g_i(x)$ *je konvexní* $(i = 1, ..., I)$ *a existuje* $\overline{x}$ *splňující* $g_i(\overline{x}) < 0$ $(i = 1, ..., I)$ *(Slaterova podmínka)*

3. *Vektory* $\nabla g_i(x^*)$, $i \in I(x^*)$, *jsou lineárně nezávislé*

První dvě nutné podmínky ve větě 6.4.1 zaručují regularitu bodu $\mathbf{x}^* \in S$, která je vyžadovaná v následující větě.

Věta 6.4.2. *Nechť jsou cílová funkce* $f(\mathbf{x})$ *a podmínky* $g_i(\mathbf{x})$ *spojitě diferencovatelné na otevřené množině* A *obsahující* S, *a nechť* $\mathbf{x}^*$ *je bodem lokálního minima takovým, že podmínky jsou v tomto bodě* $\mathbf{x}^*$ *regulární. Poté platí následující KKT podmínky:*

1. $g_i(\mathbf{x}^*) \leq 0, \quad i = 1, ..., I$

2. *existuje* $\lambda_i \geq 0$ *takové, že* $\lambda_i g_i(\mathbf{x}^*) = 0, \quad i = 1, ..., I$

3. $\nabla f(\mathbf{x}^*) + \sum_{i=1}^{I} \lambda_i \nabla g_i(\mathbf{x}^*) = 0$

Následující věta je rozšířením předcozí a uvádí nutné podmínky KKT pro úlohu 6.3, ve které dovolujeme i rovnostní podmínky $h_j(\mathbf{x})$.

Věta 6.4.3. *Nechť jsou cílová funkce* $f(\mathbf{x})$ *a podmínky* $g_i(\mathbf{x})$ *a* $h_j(\mathbf{x})$ *spojitě diferencovatelné na otevřené množině* A *obsahující* S, *nechť také* $\mathbf{x}^*$ *je bodem lokálního minima. Dále předpokládejme, že vektory* $\nabla g_i(\mathbf{x}^*)$ *a* $\nabla h_j(\mathbf{x}^*)$ *jsou lineárně nezávislé. Poté platí následující KKT podmínky:*

1. $g_i(\mathbf{x}^*) \leq 0$ $(i = 1, ..., I)$, $h_j(\mathbf{x}^*) = 0$ $(j = 1, ..., J)$.

2. *Existuje* $\lambda_i \geq 0$ $(i = 1, ...I)$ *a* $\mu_j \in \mathbb{R}$ $(j = 1, ..., J)$ *takové, že*

$$\nabla f(\mathbf{x}^*) + \sum_{i=1}^{I} \lambda_i \nabla g_i(\mathbf{x}^*) + \sum_{i=1}^{I} \mu_j \nabla h_j(\mathbf{x}^*) = 0$$

$$\lambda_i g_i(\mathbf{x}^*) = 0 \quad (i = 1, ..., I)$$

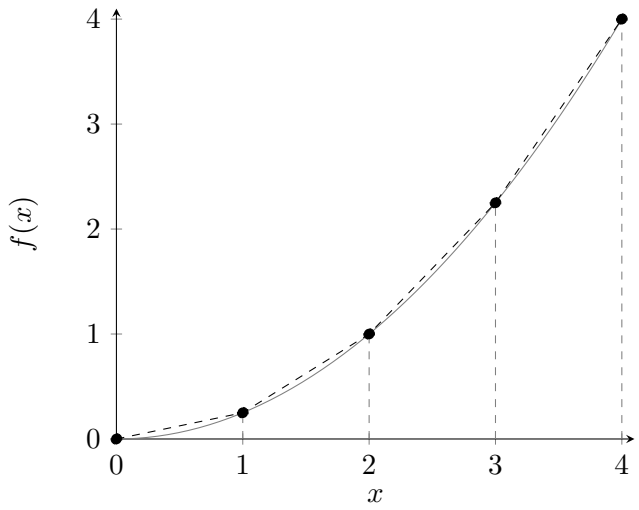
.

6.5 Separabilní programování

Tato kapitola je zpracována podle [22]. Cílová funkce $f(x_1, x_2, ..., x_n)$ je separabilní pokud může být vyjádřena jako suma n jednotných funkcí $f(x_1)$, $f(x_2)$, ..., $f(x_n)$. Tzn. můžeme si ji představit následovně

$$f(x_1, x_2, ..., x_n) = f(x_1) + f(x_2) + ... + f(x_n)$$

Myšlenka separabilního programování spočívá v rozdělení jednotlivých funkcí $f(x_1)$, $f(x_2)$, ..., $f(x_n)$ na funkce po částech lineární, abychom pak na problém mohli použít simplexový algoritmus. Speciální případ separabilního programování nastává pokud jsou podmínky $g_{ij}(x_{ij})$ konvexní pro všechna i a j,což nám zajišťuje konexní prostor řešení. Navíc pokud je $f_j(x_j)$ konvexní cílová funkce minimalizace pro všechna j, poté má problém globální řešení.



Obrázek 6.3: Po částech lineární aproximace konvexní funkce $f(x) = \frac{1}{4}x^2$

7 Toky s konvexními cenami

Úloha toku s konvexními cenami hran [23] je v základním principu stejná jako úloha minimální ceny toku. Rozdíl je v cílové funkci $f(\mathbf{x})$, která v úloze toku s konvexní cenou je nelineární. Uveďme zápis této úlohy

$$\begin{array}{ll} \text{minimalizovat} & f(\mathbf{x}) \\ \text{za podmínek:} & \mathbf{Ax} = \mathbf{b} \\ & \mathbf{O} \leq \mathbf{x} \leq \mathbf{u} \end{array} \tag{7.0}$$

kde $\mathbf{A}$ je incidenční matice sítě N a vektor $\mathbf{b}$ je vektor intenzit.

7.0.1 Po částech lineární aproximace

Jednou z metod ([23], [24]) pro řešení úlohy (7.0) je aproximace cen jednotlivých hran po částech lineární funkcí. Předpokládejme tedy, že cílová funkce $f(\mathbf{x})$ je separabilní a platí $f(\mathbf{x}) = \sum_e f_e(\mathbf{x}(e))$. Separované funkce f_e jsou cenové konvexní funkce c_e hran. Každou z těchto funkcí aproximujeme p lineárními funkcemi viz obrázek 6.3.

$$\text{minimalizovat} \sum_{e,p} c_e^p(\mathbf{x}(e)^p) \tag{7.1}$$

$$\text{za podmínek:} \sum_p \mathbf{Ax^p} = \mathbf{b} \tag{7.2}$$

$$\mathbf{O} \leq \mathbf{x^p} \leq \mathbf{u^p} \tag{7.3}$$

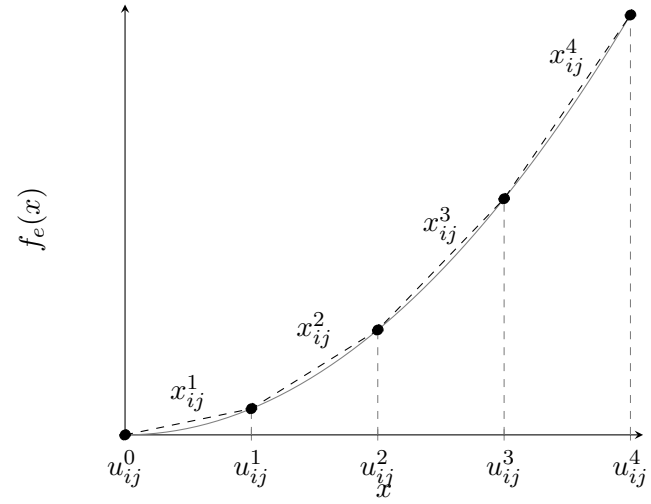
V této metodě tedy každou hranu $e = (i, j)$ nahradíme p hranami $e_{ij}^p = (i, j)$. Matice incidence $\mathbf{A}$ pro úlohu 7.1 obsahuje p-krát stejný zápis každé hrany. Každá hrana $e_{ij}^p = (i, j)$ má kapacitu $\mathbf{u}_{ij}^p = \frac{\mathbf{u}(e)}{p}$ a cenu c_{ij}^p. Touto úpravou jsme získali lineární úlohu hledání toku s minimální cenou za cenu navýšení podmínek o p-krát.

Tok v po částech lineární síti nazveme **uspořádaný** pokud jsou hrany s vyšší cenou použity poté, co jsou plně nasyceny hrany s nižší cenou. Nalezený přípustný tok nemusí být *uspořádaným* tokem, ale optimální tok je vždy *uspořádaný.*

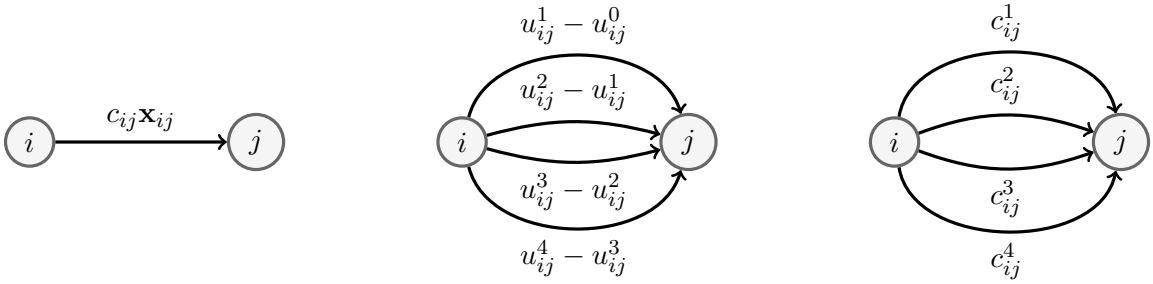
Na obrázku 7.1 je reprezentovaná linearizace cenové funkce c_{ij} po částech lineární cenovou funkcí c_{ij}^p. Vizuální reprezentace změny sítě je uvedena na příkladu jedné hrany $\mathbf{x}_{ij}$ na obrázku 7.2.

7.0.2 Frankové-Wolfeho algoritmus

Další metodou pro řešení úlohy toku v síti s konvexními cenami je použití Frankové-Wolfeho algoritmu [23]. Tato metoda řeší úlohu s konvexní diferencovatelnou cílovou funkcí a lineárními podmín-



Obrázek 7.1: Po částech linearizovaná konvexní cenová funkce hrany



Obrázek 7.2: Transformace hrany s konvexní cenou na více hran s lineárními cenami

kami. Předpokládejme, že cílová funkce $f(\mathbf{x})$ je diferencovatelná pro $\{\mathbf{x} : \mathbf{O} \leq \mathbf{x} \leq \mathbf{u}\}$ a $\nabla f(\mathbf{y})$ je gradient cílové funkce v bodě $\mathbf{y}$. Uveďme tvrzení udávající dolní omezení β cílové funkce $f(\mathbf{x})$.

Lemma 7.0.1. *Pokud je $f(\mathbf{x})$ konvexní a má spojitou první derivaci,*
tak $f(\mathbf{x}_1) \geq f(\mathbf{x}_2) + \nabla f(\mathbf{x}_2)(\mathbf{x}_1 - \mathbf{x}_2)$ pro libovolné dva body $\mathbf{x}_1$ a $\mathbf{x}_2$.

Lemma 7.0.2. *Mějme optimum $\mathbf{x}^*$ úlohy (7.0) a přípustný tok $\mathbf{y}$.*
Nechť $\mathbf{z}$ řeší úlohu $\min\{\nabla f(\mathbf{y})\mathbf{x} : \mathbf{Ax} = \mathbf{b},\ \mathbf{O} \leq \mathbf{x} \leq \mathbf{u}\}$. Potom $f(\mathbf{x}^*) \geq f(\mathbf{y}) + \nabla f(\mathbf{y})(\mathbf{z} - \mathbf{y})$.

Frankové-Wolfeho algoritmus

1. *Inicializace.* Mějme libovolný proveditelný tok $\mathbf{y}_0$. Nastavme iterace $k = 0$, dolní hranici cílové funkce $\beta = -\infty$ a zvolme ukončovací parametr $\epsilon > 0$.

2. *Řešení síťového podproblému, aktualizace mezí, kontrola ukončovací podmínky.* Označme řešení $\mathbf{z}_k$ problému $\min\{\nabla f(\mathbf{y}_k)\mathbf{x} : \mathbf{Ax} = \mathbf{r}, \mathbf{0} \leq \mathbf{x} \leq \mathbf{u}\}$. Nastavme $\beta \leftarrow \max\{l, f(\mathbf{y}_k) + \nabla f(\mathbf{y}_k)(\mathbf{z}_k - \mathbf{y}_k)\}$. Pokud $f(\mathbf{y}_k) - \beta < \epsilon$, ukončeme algoritmus s tokem $\mathbf{y}_k$ jako ϵ-optimum; jinak $k = k + 1$.

3. *Line Search.* Dovolme, aby $\mathbf{y}_k$ byl tok na úsečce mezi $\mathbf{y}_{k-1}$ a $\mathbf{z}_{k-1}$ mající nejmenší hodnotu účelové funkce a jděme na Krok 2.

Existují různé přístupy pro volbu parametru α [25]. Jedním ze způsobů volby je přes minimalizaci $\alpha = \min_{\alpha \in \langle 0,1 \rangle} f(\mathbf{y}_k + \alpha(\mathbf{z}_k - \mathbf{y}_k))$. Další možností je volba kroku α v závislosti na aktuální iteraci k: $\alpha = \frac{2}{2+k}$. Obě varianty α budou testovány v kapitole 8.

8 Algoritmy a porovnání

V této části práce porovnáme dva přístupy řešení úlohy toku s konvexními cenami uvedené v kapitole 7, které jsem implementovala pomocí softwaru Matlab.

Testování algoritmů je prováděno na souborech dimacs, které jsou vygenerované pomocí knihovny pynetgen [26] v jazyce Python. Použité soubory dimacs mají následující strukturu:

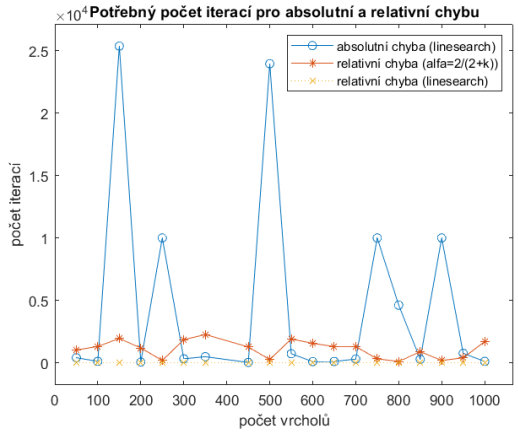
- `c` je řádek obsahující komentář. Komentáře jsou uvedeny na začátku souboru a informují o vlastnostech sítě (počet vrcholů, počet hran, počet zdrojů, ...).
- Řádky s počátečním písmenem `p` obsahují specifikaci problému, v našem případě jsou ve tvaru: `p min 100 800`. Tedy minimalizujeme síť se `100` vrcholy a `800` hranami.
- Dále tento soubor obsahuje řádky `n`, které popisují produkci nebo spotřebu jednotlivých vrcholů.
- V neposlední řadě soubor obsahuje řádky `a` popisující hrany s informacemi: počáteční vrchol hrany, koncový vrchol hrany, dolní omezení, horní omezení a cena toku.

V této práci jsou testované dva typy sítí (řídké a husté) zavedené podle předlohy v článku [17]. Řídké sítě s lineárním počtem hran uvažujeme $m = 8n$ a husté sítě se superlineárním počtem hran $m \approx n\sqrt{n}$, kde m zaokrouhlíme na celé číslo. Budeme uvažovat sítě s počtem vrcholů narůstajícím po 50 od $n = 50$ až do $n = 1000$. Řídké sítě jsou v přiložených souborech označeny jménem `dimacs_300.txt - dimacs_319.txt`. Husté sítě jsou poté pojmenované `dimacs_400.txt - dimacs_419.txt`. Budeme uvažovat sítě s konvexními cenami hran typu x^2 nebo $ax^4+bx^3+cx^2+dx$, kde koeficienty $a, b, c, d \geq 0$, aby byly ceny hran e konvexní na intervalu $\langle 0, \mathbf{u}(e)\rangle$. Poznamenejme také, že cenové funkce hran prochází 0 při nulovém toku hranou.

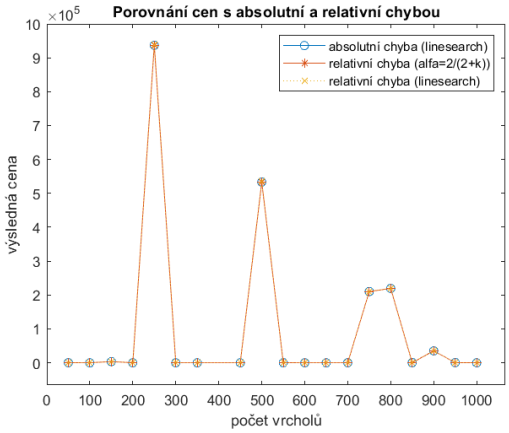
Zastavovací parametr volme $\epsilon > 0$. Uvažujme zastavovací podmínku $f(\mathbf{y_k}) - \beta < \epsilon$, kde $\epsilon = 1$, pro absolutní chybu. Pro relativní chybu uvažujme zastavovací podmínku $\frac{f(\mathbf{y_k})-\beta}{\beta} \leq \epsilon$, kde $\epsilon = 0.01$. Relativní chyba je v tomto případě vhodnějším zastavovacím parametrem, protože dopředu nevíme, v jakém řádu se pohybuje optimální cena sítě. V následující sekci otestujme jaký vliv na cenu a počet iterací má relativní a absolutní chyba použitá jako zastavovací podmínka FW algoritmu.

8.0.1 Testování absolutní a relativní chyby

Porovnejme rozdíl mezi absolutní chybou a relativní chybou na hustých sítích s konvexními cenami hran $ax^4 + bx^3 + cx^2 + dx$. Na obrázku 8.1(b) můžeme vidět, že hodnoty cen pro absolutní chybu s krokem α vypočítaným pomocí linesearch jsou skoro k nerozeznámí od cen získaných pro relativní chybu ať už s krokem α vypočítaným pomocí linesearch nebo s krokem $\alpha = \frac{2}{2+k}$. Na obrázku 8.1(a) můžeme vidět, že výpočet s absolutní chybou je výrazně náročnější na počet iterací.



(a) Porovnání počtu iterací pro ukončení pomocí absolutní chyby s $\alpha =$ linesearch a relativní chyby s $\alpha =$ linesearch a $\alpha = \frac{2}{2+k}$

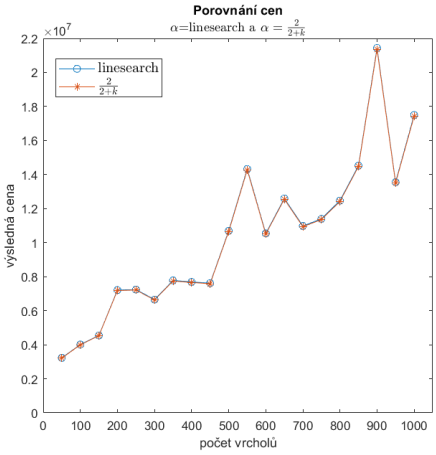


(b) Porovnání výsledných cen pro ukončení pomocí absolutní chyby s $\alpha =$ linesearch a relativní chyby s $\alpha =$ linesearch a $\alpha = \frac{2}{2+k}$

Obrázek 8.1: Husté sítě s cenou $ax^4 + bx^3 + cx^2 + dx$ na hranách

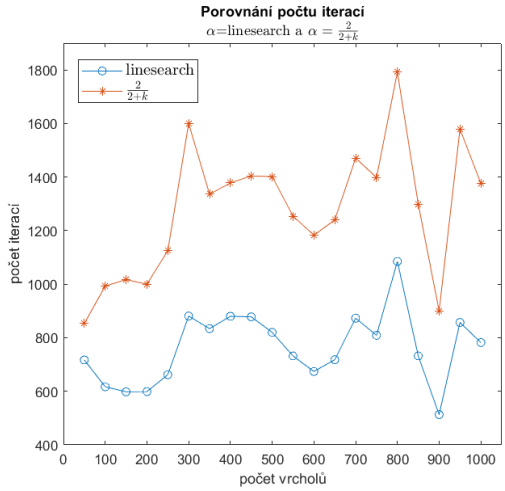
8.0.2 Testování kroku α

Porovnejme krok α v řídkých sítích s konvexní cenou hran x^2. Porovnávat budeme parametr α podle již zmíněného zdroje [25] a budeme ho volit jako $\alpha = \min_{\alpha \in \langle 0,1 \rangle} f(\mathbf{y}_k + \alpha(\mathbf{z}_k - \mathbf{y}_k))$ nebo $\alpha = \frac{2}{2+k}$. Na obrázku 8.2 můžeme vidět, že výsledné ceny toků vycházely dosti podobně.

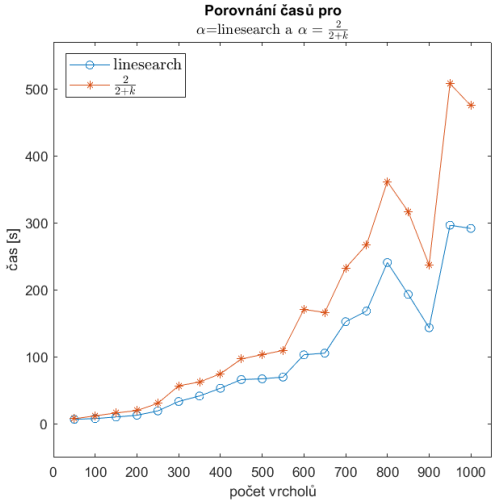


Obrázek 8.2: Řídké sítě s cenou x^2 na všech hranách

Na obrázku 8.3(a) vidíme porovnání počtu iteracích potřebných pro jednotlivé volby parametrů α a na obrázku 8.4(a) vidíme potřebný čas pro doběhnutí Frank-Wolfeho metody. Pokud zvolíme $\alpha = \frac{2}{2+k}$, tak doba výpočtu pro jednotlivé sítě zabere více času. Tedy z časového i iteračního hlediska se nám vyplatí v každém kroku napočítat krok α pomocí linesearch.



(a) Porovnání počtu iterací pro α = linesearch a $\alpha = \frac{2}{2+k}$

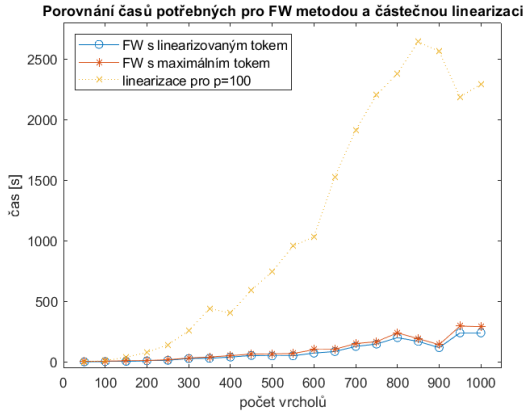


(b) Porovnání potřebných časů pro α = linesearch a $\alpha = \frac{2}{2+k}$

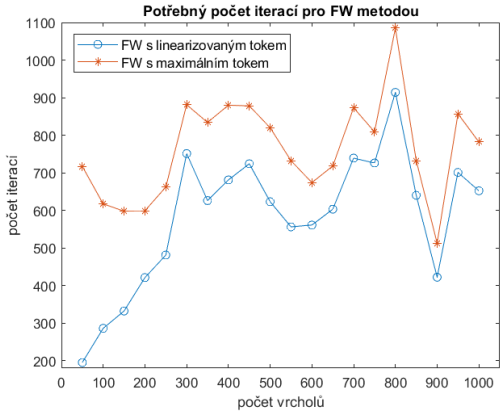
Obrázek 8.3: Řídké sítě s cenou x^2 na všech hranách

8.0.3 Porovnání volby počátečních toků ve Frankové-Wolfeho metodě

Ve snaze o optimálnější běh Frankové-Wolfeho algoritmu jsme se pokusili vložit předoptimalizovaný tok nalezený pomocí po částech lineární aproximace cen. Na obrázku 8.4(b) můžeme vidět, že Frankové-Wolfeho metoda s předoptimalizovaných tokem (FW s linearizovaným tokem) potřebuje pro nalezení optima méně iterací. Jestliže nás ale bude zajímat i časová náročnost hledání předoptimalizovaného toku, tak na obrázku 8.4(a) zjistíme, že řešení pomocí linearizace nám zabere více času, nežli samotné puštění Frankové-Wolfeho algoritmu z přípustného řešení nalezeného pomocí hledání maximálního toku v síti.



(a) Porovnání časové náročnosti pro po částech linearizovaných cenových funkcí a různých počátečních toků, vstupujících do metody FW



(b) Porovnání počtu iterací pro různé počáteční toky vstupující do metody FW

Obrázek 8.4: Řídké sítě s cenou x^2 na všech hranách

9 Závěr

V práci jsme uvedli přehled základní problematiky toků v sítích a zaměřili jsme se na klasické úlohy hledání maximálního toku a hledání toku s minimální cenou (pro případ lineárních cen).

Následně jsme se zabývali obecnější úlohou toků s konvexními cenami. Z tohoto důvodu byla do práce zařazena kapitola věnující se konvexní optimalizaci. Standardně uváděnou problematikou v této oblasti je případ konvexních kvadratických cen. My jsme se věnovali obecnější úloze, kdy cena byla dána polynomem, který je na námi použitém intervalu konvexní.

Přínosem práce je prozkoumání možností základních přístupů k úloze toků s konvexními cenami hran. Celková cena toku je součtem konvexních cen na jednotlivých hranách a tvoří tak obecnou konvexní (separabilní) funkci. Mezi zkoumané přístupy patří metoda aproximace po částech lineárními funkcemi a metoda Frankové-Wolfea. Věnovali jsme se jejich propojení a rovněž možnostem rychlého využití z praktického pohledu (pracovali jsme proto i s relativní chybou iterativní metody).

Přestože metoda Frankové-Wolfea má pro vhodná nastavení parametrů dokázanou lineární konvergenci, z praktického hlediska jsou výpočetní časy této metody značně vysoké i pro malé sítě. To byl důvod našeho přístupu pro kombinování této metody s metodou aproximací.

Metody a jejich kombinace byly testovány na řídkých a hustých sítích, tj. sítích s lineárním počtem hran, resp. superlineárním.

Možnost pro další zkoumání úlohy představuje využití tzv. metod vnitřního bodu, které jsou v oblasti toků v současné době obecně uváděny převážně jen pro úlohy s kvadratickými (konvexními) cenami.

Přílohy

Program

- `cteniSouboru.m`
- `frankWolfeAlgorithm.m`
- `generateConvexFunction.m`
- `linearizacePoCastech.m`
- `pripustnyTok.m`

Vstupní soubory

- `dimacs_300.txt - dimacs_319.txt`
- `dimacs_400.txt - dimacs_419.txt`

Generování výsledků

- `porovnani_chyb.m`
- `porovnani_kroku_alpha_x2.m`
- `porovnani_vstupu_FW_x2.m`

Bibliografie

[1] Geir Dahl. „Network flows and combinatorial matrix theory“. In: *Lecture Notes* (2010).

[2] Jørgen Bang-Jensen a Gregory Z Gutin. *Digraphs: theory, algorithms and applications*. Springer Science & Business Media, 2008.

[3] Open University Course Team. *Graphs and digraphs: Graphs 1*. en. Milton Keynes, England: Open University Worldwide, 1995.

[4] John Adrian Bondy, Uppaluri Siva Ramachandra Murty et al. *Graph theory with applications*. Sv. 290. Macmillan London, 1976.

[5] Charles A Micchelli. *Selected Papers Of Alan J Hoffman (With Commentary)*. World Scientific, 2003.

[6] Ravindra K Ahuja, Thomas L Magnanti a James B Orlin. „Network flows“. In: (1993).

[7] Jonathan L Gross, Jay Yellen a Mark Anderson. *Graph theory and its applications*. Chapman a Hall/CRC, 2018.

[8] KAMMFF Matoušek UK. *Lineární programování (Úvod pro informatiky)*. `https://iti.mff.cuni.cz/series/2006/311.pdf`. 2006.

[9] Vašek Chvátal. *Linear programming*. New York : W.H. Freeman, 1983.

[10] Milan Hladík. „Celočíselné Programování“. In: *Katedra aplikované matematiky* (2017).

[11] Tim Roughgarden. *CS261: A Second Course in Algorithms Lecture# 1: Course Goals and Introduction to Maximum Flow*. 2016.

[12] L. R. Ford a D. R. Fulkerson. *Flows in Networks*. Princeton University Press, 1962.

[13] William J. Cook et al. *Combinatorial optimization*. USA: John Wiley & Sons, Inc., 1998. ISBN: 047155894X.

[14] Martin Mareš a Tomáš Valla. *Průvodce labyrintem algoritmů*. CZ. NIC, 2017.

[15] Bernhard H Korte et al. *Combinatorial optimization*. Sv. 1. Springer, 2011.

[16] Andrew V. Goldberg a Robert E. Tarjan. „A new approach to the maximum-flow problem“. In: *J. ACM* 35.4 (říj. 1988), s. 921–940. ISSN: 0004-5411. DOI: `10.1145/48014.61051`. URL: `https://doi.org/10.1145/48014.61051`.

[17] Péter Kovács. „Minimum-cost flow algorithms: an experimental evaluation“. In: *Optimization Methods and Software* 30.1 (2015), s. 94–127.

[18] Sennosuke Watanabe, Hodaka Tanaka a Yoshihide Watanabe. „Network Simplex Algorithm associated with the Maximum Flow Problem“. In: *arXiv preprint arXiv:1706.04302* (2017).

[19] Nguyen V. Thoai Reiner Horst Panos M. Pardalos. *Introduction to Global Optimization*. Kluwer Academic Publisher, 1995.

[20] Marc Peter Deisenroth, A Aldo Faisal a Cheng Soon Ong. *Mathematics for machine learning*. Cambridge University Press, 2020.

[21] Stephan Wolf, Stephan M Günther a M Sc. „An introduction to duality in convex optimization“. In: *Network* 153 (2011), s. 153–162.

[22] Hamdy A Taha. „Operations Research An Introduction“. In: (2017).

[23] Jeff L Kennington a Richard V Helgason. *Algorithms for network programming*. John Wiley & Sons, Inc., 1980.

[24] F.-Javier Heredia. *Convex Cost Flows*. `https://upcommons.upc.edu/bitstream/handle/2117/191052/nf07_convex_cost_flows-4969.pdf?sequence=7&isAllowed=y`.

[25] Jérôme Bolte, Cyrille W Combettes a Edouard Pauwels. „The Iterates of the Frank–Wolfe Algorithm May Not Converge“. In: (2022). URL: `https://hal.archives-ouvertes.fr/hal-03579585`.

[26] *PyNETGEN*. `https://pypi.org/project/pynetgen/`. 2022.