ZÁPADOČESKÁ UNIVERZITA V PLZNI
FAKULTA APLIKOVANÝCH VĚD
KATEDRA MATEMATIKY

BAKALÁŘSKÁ PRÁCE

DYNAMICKÉ TOKY V SÍTÍCH

PLZEŇ 2017

PETERKA DAVID

Čestné prohlášení

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím odborné literatury a pramenů uvedených v seznamu na konci práce.

V Plzni dne ....

vlastnoruční podpis

Poděkování

Rád bych chtěl poděkovat vedoucímu práce Doc. Ing. Romanu Čadovi, Ph.D. za cenné rady *a veškerý* čas, který mi věnoval při konzultacích. Dále bych také chtěl poděkovat moji rodině za veškerou podporu po celou dobu studia.

Abstrakt

Tato bakalářská práce je zaměřena na hledání optimálního řešení v některých situacích, které jsou modelovány pomocí ohodnoceného grafu. Hlavním cílem této práce je poskytnout přehled algoritmů pro hledání maximálního, resp. optimálního toku v síti jak v klasické statické formulaci tak *v dynamické* podobě. Dále je v práci porovnána výpočetní složitost úloh ve statické *a dynamické* podobě. V poslední řadě je práce zaměřena na vybrané aplikace těchto úloh *v ekonomii*.

Klíčová slova

ohodnocený graf, statické toky v síti, minimální cena toku, maximální tok, dynamické toky *v síti*, algoritmy

Abstract

This bachelor thesis is focused on finding optimal solutions in some situations that are modeled using a weighted graph. The main purpose of this bachelor thesis is to provide an overview of algorithms for searching for the maximum, respectively optimal flow in the network both in classical static formulation and in dynamic form. Further in the work, there is a comparison of computational complexity of the tasks in a static and dynamic versionFinally, the work is focused on selected applications of these tasks in economics.

Keywords

weighted graph, static flows in networks, minimum cost flow, maximum flow, dynamic flows in networks, algorithms

Obsah

1 Úvod

V práci se budeme zabývat hledáním optimálního řešení v situacích, které jsou modelovány pomocí ohodnoceného grafu. V praxi se můžeme setkat s celou řadou rozhodovacích situací, které lze modelovat právě na základě ohodnoceného grafu. Hlavní úlohou, kterou se budeme zabývat je nalezení maximálního, resp. optimálního toku v síti. Dalším cílem práce je uvést algoritmy pro řešení problému této úlohy.

V kapitole 2 je z počátku uvedena historie teorie grafů a následně je v teorii grafů uveden vznik toků v sítích.

Kapitola 3 a 4 je zaměřena na určení základních pojmů a tvrzení o grafech a tocích, které jsou důležité pro osvojení problematiky v následujících kapitolách. Tvrzení nejsou uváděny *s důkazy* avšak důkazy lze nalézt v odkazované literatuře.

Kapitola 5 je zaměřena na problematiku hledání maximálního toku ve statické formulaci. *V kapitole* se zabýváme časově neměnnými toky a algoritmy, které slouží k nalezení maximálního toku v síti. Některé algoritmy jsou dokonce ilustrovány v grafickém znázornění. Dále se v kapitole zabýváme výpočetní složitostí jednotlivých algoritmů a nakonec kapitoly je uveden pojem cirkulace, který bude užitečný pro následující kapitoly. Tvrzení, která jsou zde zmíněna jsou uvedeny opět bez důkazů, stejně tak v kapitole 6 a 7. Důkazy lze však opět najít v odkazované literatuře.

V kapitole 6 jsou uvedeny algoritmy pro určení optimálního toku s minimální cenou. Opět se zde zabýváme časově neměnnými toky. Dané algoritmy jsou popsány v jednotlivých krocích a opět je u algoritmů uvedena jejich výpočetní složitost.

Kapitola 7 je zaměřena na hledání maximálního toku v dynamické síti. Zde se tedy zabýváme toky, které popisují strukturovanou síť a rozhodovací problémy v průběhu času. Hledání maximálního dynamického toku je ilustrováno graficky. Dále jsou uvedeny techniky pro řešení problému dynamického toku a je zde zmíněna výpočetní složitost daných problémů týkajících se hledání dynamických toků. Na závěr je v kapitole shrnuta a porovnána výpočetní složitost ve statické formulaci oproti dynamické.

V kapitole 8 si uvedeme uplatnění toků v ekonomii. Nejdříve se budeme zabývat modelováním dopravní sítě pomocí teorie her, kde bude zmíněn tzv. Braesův paradox. Dále bude v této kapitole popsána optimalizace letového provozu a problémy týkající se letového provozu.

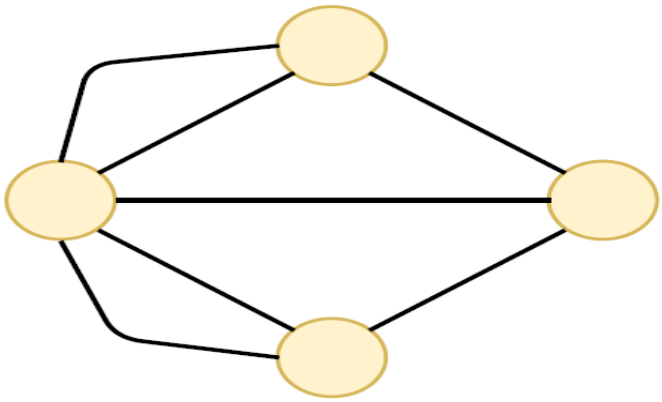
V závěrečné kapitole 9 je shrnuta celá práce a jsou zde okomentovány veškeré výsledky *a zmíněny* náměty pro případné rozšíření práce.

Přehled značení

G	spojitý orientovaný graf
V	množina vrcholů
E	množina hran
c	funkce určující kapacitu hrany
s	speciální vrchol sítě (zdroj)
t	speciální vrchol sítě (spotřebič)
$\mathbb{R}$	reálná čísla
f	funkce určující tok
G_f	reziduální (rezervní) síť
G_L	čistá síť
c_f	rezervní kapacita
$r(e)$	rezerva hrany
$r^+(v)$	rezerva hran vstupujících do vrcholu
$r^-(v)$	rezerva hran vystupujících z vrcholu
$r(v)$	rezerva vrcholu, minimum z $r^+(v)$ a $r^-(v)$
$h(v)$	výška vrcholu
$f(v)$	přebytek vrcholu v - značeno také jako $excess_f(v)$
τ	přepravní čas
$imbal_f(v)$	nerovnováha vrcholu v
$G' \sqsubseteq G_f$	graf G' je podgrafem grafu G_f
$deg(v)$	stupeň vrcholu ($deg^+(v)$ - vstupní, $deg^-(v)$ - výstupní)

2 Historie teorie grafů a vzniku toků v sítích

Za zakladatele teorie grafů je považován Leonhard Euler. V roce 1736 zveřejnil řešení příkladu Sedmi mostů města Královce. Zadání úlohy spočívalo v tom, jestli je možné projít každým mostem ve městě právě jednou a vrátit se zpět do původního místa. Euler převedl tuto situaci na graf tak, že si každý břeh představil jako vrchol a každý most použil jako hranu, která břehy spojuje. Matematicky dokázal, že úloha není řešitelná. Na následujícím obrázku je graf, který zachycuje úlohu 7 mostů města Královce.



Obrázek 1: Graf sedmi mostů města Královce

Vznik teorie grafů jako samostatné matematické disciplíny je možno počítat od roku 1936, kdy vyšla monografie D. Königa „Teorie konečných a nekonečných grafů“. Od té doby vyšla řada publikací, které se dále zabývají nejrůznějšími aplikacemi teorie grafů v oblasti matematiky, informatiky, ekonomie, atd. V řadě případů je znázornění pomocí grafů velmi výhodné.

V roce 1845 publikoval Gustav Kirchhoff zákony, které platí v elektrických obvodech a slouží k výpočtu napětí a proudu v jednotlivých větvích obvodu. V teorii grafů našly své uplatnění při studiu tzv. toků v sítích. Problémy řešené v rámci zkoumání toků v sítích jsou motivovány praktickými úlohami o propustnosti - typickým příkladem je propustnost železniční, silniční nebo vodovodní sítě. Více o vzniku a vývoji teorie grafů lze nalézt zde [30].

3 Základní pojmy v teorii grafů

Grafy si můžeme představit jako zjednodušení skutečného světa, kdy daný problém lze znázornit pomocí bodů a čar, které je spojují a na základě toho tím udávají jejich vlastnosti. V teorii grafů se těmto bodům říká **vrcholy** (uzly) a čáry, které je spojují jsou nazývány jako **hrany** grafu. Těmto schématům potom říkáme grafy. Pro pochopení problematiky v následujících kapitolách si nejprve zadefinujeme základní pojmy v teorii grafů (viz. [30], [9, str. 45-51], [19, str. 13-20], [8] a [16]).

3.1 Definice základních pojmů

3.1.1 Definice. Obecný graf G je uspořádaná trojice (V, E, ϵ), kde

- V je neprázdná a konečná množina uzlů,
- E je konečná množina hran a
- ϵ je incidenční zobrazení $\epsilon : \mathrm{E} \to V^2 \cup \binom{V}{2}$, které přiřazuje každé hraně e $\in$ E uspořádanou dvojici uzlů $(u, v) \in V^2$ a dvojici uzlů $\{u, v\} \in \binom{V}{2}$.

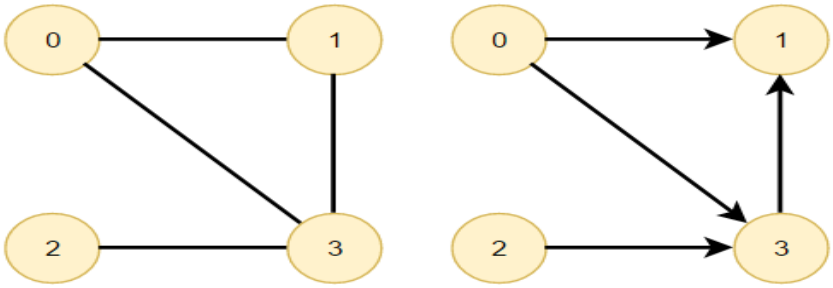
3.1.2 Poznámka. O uzlech spojených hranou říkáme, že spolu sousedí a nazýváme je tedy sousední uzly. Ve vztahu uzel - hrana se používá pojem **incidence.** Uzel, který není incidentní s žádnou hranou se nazývá **izolovaný uzel.**

3.1.3 Alternativní definice. Graf G je uspořádaná dvojice (V, E), kde V je neprázdná konečná množina a E je množina dvoubodových podmnožin množiny V. Prvky z množiny V se nazývají vrcholy grafu G a prvky z množiny E $\subseteq \binom{V}{2} \cup (V \times V)$ se nazývají hrany grafu G, přičemž množina $\binom{V}{2}$ značí neorientované hrany a množina $V \times V$ značí orientované hrany. Zkratky V a E pocházejí z anglického jazyka, kde vrcholy jsou anglicky Vertices a hrany Edges.

3.1.4 Poznámka. Orientovanou hranou značíme dvojici (u, v), kde u nazýváme předchůdcem vrcholu v a v nazýváme následovníkem vrcholu u. Dále orientované hrany typu (u, u) a neorientované hrany typu $\{u, u\}$ nazýváme smyčkou. Grafy, které obsahují smyčky se nazývají **pseudografy.**

3.1.5 Definice. Graf G je neorientovaný, pokud $E \subseteq \binom{V}{2}$. Potom tedy hrana $e = \{u, v\}$ je neorientovaná hrana a spojuje vrcholy u a v. Hrana je vždy znázorněna úsečkou.

3.1.6 Definice. Graf G je orientovaný, pokud $E \subseteq V \times V$. Potom tedy hrana $e = (u, v)$ je orientovaná hrana, která má počátek ve vrcholu u a konec ve vrcholu v. Tato hrana je znázorněna šipkou.



Obrázek 2: Neorientovaný a orientovaný graf

3.1.7 Poznámka. Ve výše uvedených grafech mluvíme o prostých grafech. Prostým grafem označujeme graf, který neobsahuje násobné hrany (více hran spojujících stejné vrcholy).

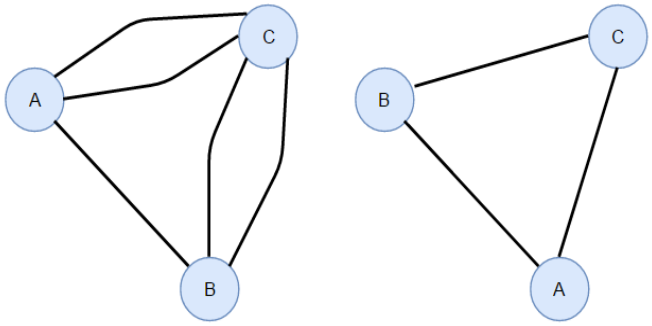
Grafy můžeme dále klasifikovat na základě:

1. **Násobnosti hran**

   a) Jednoduchý graf (orientovaný, neorientovaný) je graf, který neobsahuje žádné násobné hrany ani smyčky.

   b) Prostý graf (orientovaný, neorientovaný) je graf, který má násobnost každé hrany mezi dvěma různými vrcholy nejvýše 1.

   c) Multigraf je graf (orientovaný, neorientovaný), který obsahuje hranu jejíž násobnost je větší než 1. V tomto případě se tedy nejedná o prostý graf.



Obrázek 3: Multigraf a prostý graf

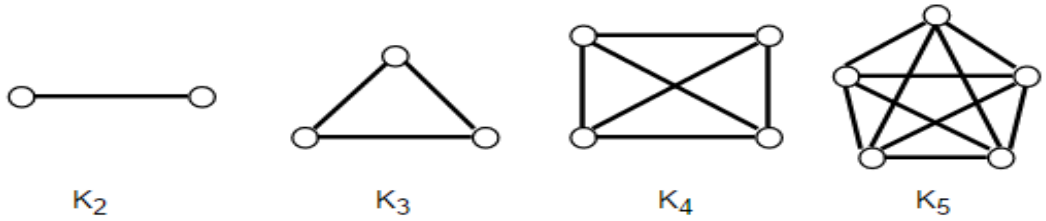
2. **Podle počtů hran a uzlů**

   a) Konečný graf je graf, který obsahuje konečný počet hran a uzlů.

   b) Nekonečný graf je graf, který obsahuje nekonečný počet uzlů.

   c) Prázdný graf je graf, který nemá žádný vrchol ani žádnou hranu. Můžeme tedy říci, že $V = \emptyset$ a $E = \emptyset$.

   d) Úplný graf je graf, kde každé dva vrcholy jsou spojeny právě jednou hranou. Úplný graf na **n** vrcholech značíme K_n. Pro $n \geq 1$ platí, že $E(K_n) = \binom{V}{2}$.



Obrázek 4: Úplné grafy pro $n \in (1, 6)$

3.1.9 Definice. Graf G se nazývá ohodnocený (orientovaný, neorientovaný), pokud ke každé hraně grafu G je přiřazena váha $w(e) \in \mathbb{R}$.

3.1.10 Definice. Stupeň vrcholu u je značen jako $deg(u)$. Stupeň vrcholu rozlišujeme na základě toho zda se jedná o orientovaný nebo neorientovaný graf.

1. U neorientovaného grafu lze definovat stupeň vrcholu jako počet hran, které s daným vrcholem incidují.

2. U orientovaného grafu rozlišujeme, zda se jedná o vstupní stupeň vrcholu $deg^+(u)$ nebo *o výstupní* stupeň vrcholu $deg^-(u)$. Celkový stupeň vrcholu je tedy definován jako součet vstupních a vystupujících hran.

   - Vstupní stupeň $deg^+(u)$ je definován jako:

   $$deg^+(u) = |\{e \in E \mid \exists v \in V : e = (v, u)\}|$$

   - Výstupní stupeň vrcholu $deg^-(u)$ je definován jako:

   $$deg^-(u) = |\{e \in E \mid \exists v \in V : e = (u, v)\}|$$

V grafech se často přesouváme z jednoho uzlu do druhého. V takovém případě je nutné definovat pojmy, které souvisejí s posunem z jednoho uzlu do druhého a dále.

a) Sled v grafu G je uspořádaná posloupnost uzlů a hran, kde je možné aby se uzly a hrany opakovaly. Jestliže skončíme při průchodu grafem v uzlu ve kterém jsme začali jedná se *o* ***uzavřený sled.*** Pokud se skončí v jiném uzlu než v počátečním mluvíme o **otevřeném sledu.**

b) Tah v grafu G je sled, kde se žádná hrana nevyskytuje více než jednou. Opět se dělí na otevřený nebo uzavřený tah.

c) Cesta v grafu G je otevřený tah, při kterém se každý vrchol vyskytuje pouze jednou.

d) Kružnice v grafu G je uzavřená neorientovaná cesta. V orientovaném grafu se zavádí pojem cyklus. Pokud graf neobsahuje žádný cyklus nazývá se **acyklický**.

3.1.11 Definice. Graf G se nazývá souvislý pokud mezi všemi dvojicemi uzlů existuje alespoň jedna cesta. V opačném případě mluvíme o nesouvislém grafu.

4 Toky v sítích

Toky v sítích jsou především využívány v oblasti dopravních problémů, kde se snažíme maximalizovat využití přepravní kapacity a to na základě minimalizace přepravních nákladů. Hledání maximálního toku se v praxi využívá nejčastěji při plánování rozvodu vody nebo kanalizace, ale lze využít i například při pohybu dat po internetové síti. Pro hledání maximálního toku existují základní typy algoritmů, kterými se budeme zabývat v dalších kapitolách práce. Nejprve je třeba si zadefinovat základní pojmy týkající se této kapitoly pro hledání optimálního řešení maximálního toku (viz. [9, str. 145-147], [22, str. 7-10] a [7, str. 108-110]).

4.1 Základní pojmy

1. Síť S je uspořádaná čtveřice (G, s, t, c), kde $G = (V, E, \epsilon)$ je orientovaný konečný ohodnocený graf. Síť obsahuje dva speciální vrcholy s a t, kde $s, t \in V$, přičemž s nazýváme zdrojem (source) a t nazýváme spotřebičem neboli stokem (target). Každá hrana v síti e má přidělenou kapacitu $c(e)$, kde kapacita je funkce $c : E \to \mathbb{R}_0^+$.

2. Tok v síti je funkce $f : E \to \mathbb{R}_0^+$, která splňuje následující podmínky

   - $0 \leq f(e) \leq c(e)$ pro každou hranu $e \in E$,
   - $\sum_{uv \in E} f(uv) = \sum_{vu \in E} f(vu)$ pro každý vrchol $v \in V \setminus \{s, t\}$.

   První podmínka udává, že průtok hranou $f(e)$ je nezáporný a musí být vždy menší nebo roven kapacitě hrany $c(e)$. Druhá podmínka udává, že co do daného vrcholu přiteče, tak z něj musí také odtéci. Druhá podmínka je známá také jako zákon zachování toku, jelikož se ve vrcholech tok neztrácí. Druhé podmínce se ve fyzice říká **Kirchhoffův zákon**. Tok, který vede ze zdroje s do spotřebiče t se označuje jako (s, t)-tok.

3. Bilance vrcholu $f(v)$ neboli přebytek (někdy značeno také jako $excess_f(v)$) ve vrcholu je určen rozdílem mezi přítokem do vrcholu v a odtokem z vrcholu v. Tento přebytek je určen následujícím vzorcem

$$f(v) = \sum_{uv \in E} f(uv) - \sum_{vu \in E} f(vu) \tag{1}$$

4. Velikost toku $|f|$ lze označit jako množství, které protéká ze zdroje s do spotřebiče t. Ze zákona zachování toku víme, že se tok ve vrcholech neztrácí, tedy velikost toku $|f|$ lze vypočítat jako tok, který přitéká do spotřebiče tzn. $|f| = f(t)$. Velikost toku lze měřit *i jiným* způsobem a to jako velikost toku, který vytéká ze zdroje. Platí tedy, že

$$| f | = \sum_{sv \in E} f(sv) = \sum_{vt \in E} f(vt). \tag{2}$$

5. Řez mezi zdrojem s a spotřebičem t se nazývá množina všech hran R, které spojují uzly množiny V_1 s uzly množiny V_2, kdy V_1 je množina uzlů, která obsahuje počáteční uzel a všechny uzly dosažitelné z počátečního uzlu po nenasycené cestě. Množina uzlů V_2 je taková množina, která obsahuje koncový uzel a všechny zbývající uzly. Kapacitu řezu definujeme jako $c(R) = \sum_{e \in R} c(e)$. Minimální řez je řez, který má nejmenší kapacitu.

6. Rezervní kapacita hrany je určena rozdílem kapacity a toku na dané hraně. Minimum rezervních kapacit hran na zvolené cestě se nazývá **rezervní kapacita cesty**. Podgraf tvořený pouze hranami s kladnou rezervní kapacitou se nazývá **rezervní síť** (neboli reziduální síť) označována jako G_f.

7. Rezervní síť (reziduální síť) je podgraf tvořený pouze hranami s kladnou rezervní kapacitou a je definována jako $G_f = (V, E_f)$, kde

$$E_f := \{(u, v) \in V \times V |\, c_f(u, v) > 0\}, \tag{3}$$

kde rezervní kapacita $c_f(u, v)$ je funkce $V \times V \to \mathbb{R}_0^+$, která je pro každou dvojici vrcholu $(u, v) \in V$ definována následovně

$$c_f(u, v) := \begin{cases} c(u, v) - f(u, v) & pokud\ (u, v) \in E, \\ f(v, u) & pokud\ (v, u) \in E, \\ 0 & jinak. \end{cases} \tag{4}$$

8. Nenasycená (**vylepšující**) polocesta P je neorientovaná cesta v rezervní síti z vrcholu s do vrcholu t, tj. posloupnost navazujících hran $e_1, e_2, \ldots, e_m$, kde ve směru ze zdroje do spotřebiče platí, že $f(e_i) < c(e_i)$ a v opačném směru platí, že $f(e_i) > 0$. Nenasycená polocesta P je tedy cesta s kladnými rezervami kapacit všech hran.

9. Cesta P ze zdroje do spotřebiče tj. $P = s, v_1, v_2, \ldots, v_i, t$ se nazývá **nasycená** pokud pro nějakou hranu $e_i = (v_{i-1}, v_i)$ orientovanou po směru platí, že $f(e_i) = c(e_i)$ nebo pro nějakou hranu $e_i = (v_i, v_{i-1})$ orientovanou proti směru platí, že $f(e_i) = 0$. Jestliže je každá cesta ze zdroje do spotřebiče nasycená, potom je tok nasycený. Pokud cesta není nasycená, jedná se o nenasycenou (vylepšující) cestu, kde pro každou hranu platí, že rezervní kapacita je větší než 0. Cesta ze zdroje do spotřebiče je označována jako (s, t)-cesta.

10. Rezerva hrany je určena na základě toho zda se jedná o dopřednou nebo zpětnou hranu. Dopřednou hranou označujeme hranu, která směřuje od zdroje ke spotřebiči ve směru cesty. Zpětnou hranou nazveme hranu, která směřuje proti směru cesty. Množství toku, které je možné poslat po směru hrany nazýváme rezervou po směru hrany e a vypočteme ji vztahem

$$r(e) = c(e) - f(e). \tag{5}$$

Množství toku, které je možné poslat proti směru hrany se nazývá rezerva proti směru hrany e a její velikost je $f(e')$. Pokud budeme uvažovat, že ke každé hraně existuje hrana opačná potom rezervu hrany $e = (u, v)$ při toku f definujeme jako

$$r(e) = (c(e) - f(e)) + f(e'), \tag{6}$$

kde $e' = (v, u)$. V případě, že nebude k nějaké hraně existovat opačná hrana, přidáme opačnou hranu, které přiřadíme nulovou kapacitu.

4.1 Věta. [9, str. 147] Pokud v rezervní síti existuje vylepšující polocesta P ze zdroje do spotřebiče, potom tok f není maximální.

Z věty (4.1) lze říci, že velikost toku v síti je maximální pokud v rezervní síti neexistuje vylepšující polocesta.

4.2 Věta. [9, str. 146] Pro každý (s,t)-tok f a každý (s,t)-řez R platí, že $|f| \leq c(R)$. Velikost maximálního toku je nejvýše rovna velikosti minimálnímu řezu.

V každé síti existuje maximální tok a minimální řez. Velikost jakéhokoliv (s,t)-řezu lze považovat za horní odhad velikosti toku. Následující věta, která je jednou hlavních vět o tocích byla objevena Ford Fulkersonem v roce 1956.

4.3 Věta. [9, str. 146] Pokud existuje maximální (s,t)-tok, potom velikost tohoto toku *v síti* je rovna velikosti minimálnímu řezu. Pokud f je maximální (s,t)-tok, potom v síti neexistuje vylepšující cesta a tedy existuje řez (s,t), takový že platí

$$max\left\{|f| : f\ je\ (s,t) - tok\right\} = min\left\{c(R) : c(R)\ je\ (s,t) - rez\right\}. \tag{7}$$

4.4 Poznámka. V celé kapitole o tocích se budeme pro jednoduchost zabývat pouze orientovanými grafy. Pokud bychom chtěli hledat maximální tok nebo minimální řez v neorientovaném grafu museli bychom nahradit každou neorientovanou hranu dvojicí šipek jdoucích proti sobě. Touto operací bychom vytvořili orientovaný graf, kde kapacity obou šipek budou stejné jako kapacita původní hrany.

.

5 Toky ve statické podobě

V této kapitole si uvedeme několik základních algoritmů ve statické podobě tzn. časově neměnné toky pro hledání maximálního toku v síti. Problém maximálního toku spočívá v tom, že se vždy snažíme najít maximální tok, který lze odeslat z počátečního vrcholu do koncového vrcholu, pokud na linkách (hranách) mezi vrcholy existují kapacity, které nelze překročit. Hodnotu maximálního toku lze zkoumat například v silniční dopravě, kdy kapacity na linkách odpovídají počtu aut, které danou cestou projedou za jednotku času (minutu). Využití lze nalézt také *v datové* síti nebo pro přípravu evakuačních plánů. Mezi základní algoritmy patří *Ford - Fulkersonův*, Dinicův / Edmonds - Karpův, Metoda tří Indů, Škálovací a nakonec Goldbergův. Nyní si popíšeme fungování jednotlivých algoritmů pro hledání maximálního toku (viz. [17], [14], [9, str. 148-160], [18], [26] a [19, str. 164-170]).

5.1 Ford - Fulkersonův algoritmus

Tento algoritmus je určen pro hledání maximálního toku a byl zmíněn v roce 1957. Hlavním cílem je vždy nalézt cestu ze zdroje do spotřebiče takovou, která není nasycená. Tento krok opakujeme do té doby dokud neexistuje žádná nenasycená (s, t)-cesta. Algoritmus si nyní uvedeme v jednotlivých krocích:

1. V daném ohodnoceném grafu v prvním kroku přiřadíme každé hraně tok $f(e) = 0$.

3. Z původní sítě vytvoříme rezervní síť.

3. Ve třetím kroku se snažíme najít v rezervní síti vylepšující polocestu ze zdroje do spotřebiče. Ze začátku tedy zvolíme libovolnou polocestu (bez opakujících se vrcholů). Dále určíme nejmenší rezervu ϵ hrany na zvolené polocestě P. Následně na hranách ve směru cesty zvýšíme tok o ϵ. Nejmenší rezervu získáme jako

$$\epsilon := min_{e \in P} \{c(e) - f(e)\}. \tag{8}$$

Pokud by však vylepšující cesta obsahovala zpětné hrany, tok se na těchto hranách sníží *o* ϵ. Nejmenší rezervu ϵ hrany na zvolené polocestě P, která obsahuje i zpětné hrany získáme jako

$$\epsilon := min_{e \in P} \{c(e) - f(e), f(e')\}. \tag{9}$$

Nový tok f' je dán následovně

$$f'(e) := \begin{cases} f(e) + \epsilon & pokud\ e \in P, \\ f(e') - \epsilon & pokud\ e' \in P, \\ f(e) & pro\ e\ jinak. \end{cases} \tag{10}$$

Každé hraně se také změní hodnota kapacity $c(e)$ a to následovně

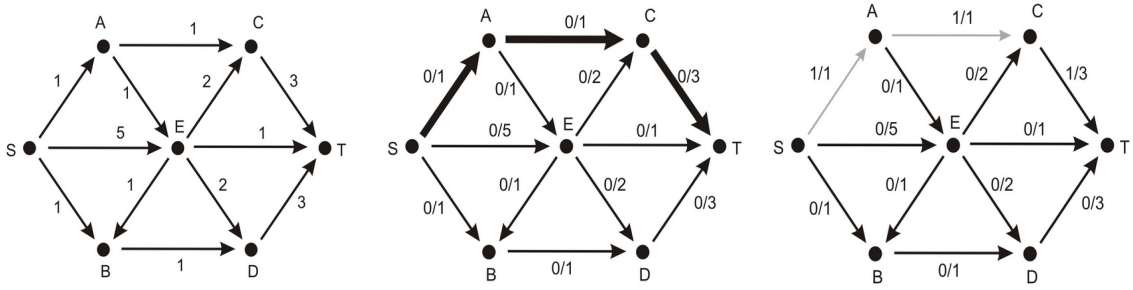
$$c'(e) := \begin{cases} c(e) - \epsilon & pokud\ e \in P, \\ c(e') + \epsilon & pokud\ e' \in P, \\ c(e) & pro\ e\ jinak. \end{cases} \tag{11}$$

4. V dalším kroku aktualizujeme rezervní síť a hledáme opět vylepšující polocestu ze zdroje do spotřebiče přes hrany u kterých platí, že $f(e) < c(e)$ nebo $f(e') > 0$. Tento krok opakujeme do doby dokud existuje nějaká vylepšující polocesta, tzn. že existuje nenasycená polocesta.

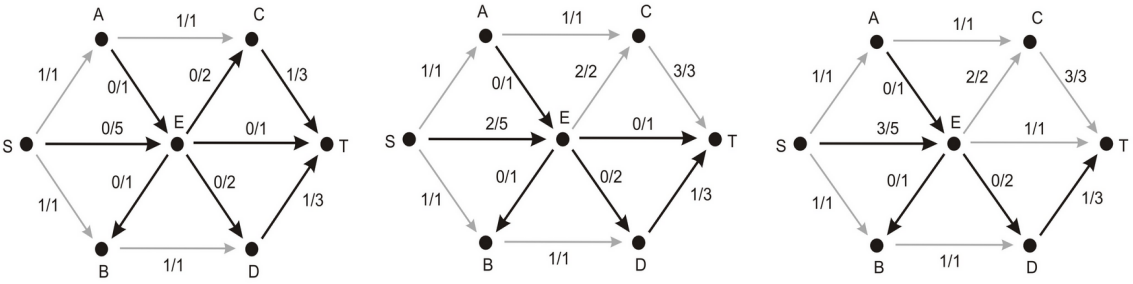
5. Pokud již neexistuje žádná nenasycená polocesta ze zdroje do spotřebiče algoritmus se zastaví a je nalezen maximální tok, což nám říká věta (4.1). Maximální tok je potom tvořen součtem rezerv ϵ, které byly postupně spočteny při hledání vylepšujících polocest v rezervní síti.

5.1.1 Příklad Ford Fulkersonova algoritmu

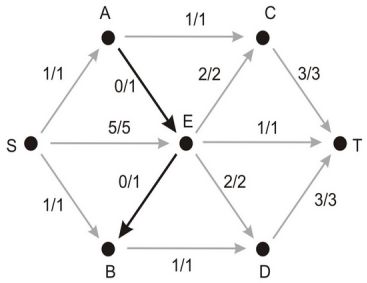
V příkladech pro hledání maximálního toku se budeme zabývat situacemi, kdy síť obsahuje pouze jeden zdroj a jeden stok a hrany mají celočíselné kapacity, tedy hodnota maximálního toku bude celočíselná, jelikož v průběhu činnosti algoritmu se pouze sčítá. Sečtením celočíselných kapacit dostáváme ve výsledku tedy celočíselnou hodnotu toku (viz. [7, str. 112 Věta 3.5.5]). Pokud má síť celočíselné kapacity stoupne velikost toku minimálně o jedna a algoritmus se zastaví po nejvýše tolika krocích, kolik je nějaká horní mez pro velikost maximálního toku, což může být například součet kapacit všech hran vedoucích do stoku. V případě, že by síť obsahovala racionální kapacity, musíme tyto kapacity převést na celočíselné pomocí přenásobením nejmenšího společného jmenovatele a poté řešit daný algoritmus. Hodnota maximálního toku se na konci vydělí nejmenším společným jmenovatelem a tedy hodnota toku bude racionální. I v tomto případě se algoritmus zastaví. V síti s iracionálními kapacitami Ford-Fulkersonův algoritmus skončit nemusí.



Obrázek 5: Hledání vylepšující cesty pomocí Ford Fulkersonova algoritmu [31]



Obrázek 6: Hledání vylepšující cesty pomocí Ford Fulkersonova algoritmu [31]



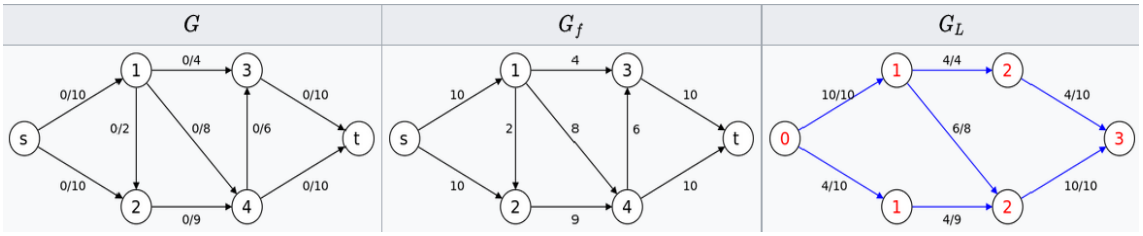
Obrázek 7: Výsledná síť, ve které již neexistuje vylepšující (s, t)-cesta [31]

5.2 Dinicův / Edmonds - Karpův algoritmus

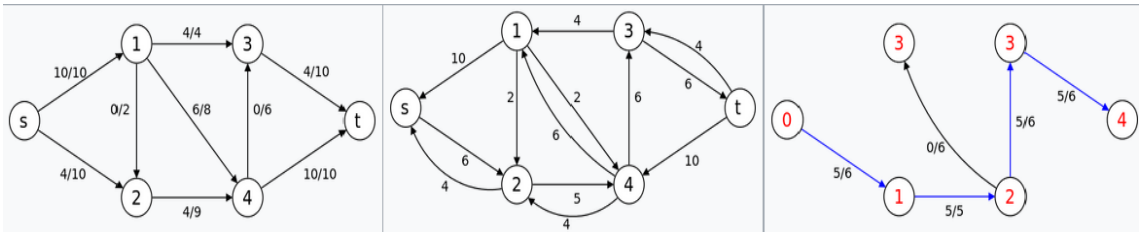
Tento algoritmus je opět určen pro hledání maximálního toku. Jedná se o zefektivnění Ford Fulkersonova algoritmu (1972). Algoritmus se snaží vyhledat cestu ze zdroje do spotřebiče pomocí co nejmenšího počtu hran, tj. pomocí prohledáváním sítě do šířky. Kroky daného algoritmu jsou určeny následujícím pořadím:

1. V ohodnoceném grafu přiřadíme každé hraně nulový tok tzn. $f(e) = 0$.
2. Z původní sítě vytvoříme síť rezerv. Síť rezerv k toku f představuje síť, kde kapacity hran jsou tvořeny rezervami hran z původní sítě.
3. V síti rezerv najdeme nejkratší (s, t)-cestu. Pokud nelze takovou cestu najít algoritmus je ukončen a je nalezen maximální tok.
4. Dále provedeme **pročištění sítě.** Zachováme pouze hrany a vrcholy, které tvoří nejkratší (s, t)-cestu. Vytvoříme tzv. „čistou síť“.
5. Nalezneme **blokující tok** v čisté síti. Blokujícím tokem se myslí tok, kdy na každé orientované cestě ze zdroje do spotřebiče existuje alespoň jedna hrana, kde je tok roven kapacitě. Z počátku je blokující tok roven nule.
6. V čisté síti nalezneme nejkratší (s, t)-cestu a následně na hranách ve směru cesty zvýšíme tok o ϵ, což je nejmenší rezerva hrany na cestě. Hrany, které vzniknou jako nasycené odstraníme a dočistíme síť. Po smazání nasycených hran můžou v síti vzniknout tzv. „slepé uličky.“ Na základě smazání nasycené hrany může dojít k znečištění sítě. Dočištění sítě se provádí následujícím způsobem:
   a) Vrcholy se rozdělí do jednotlivých vrstev na základě vzdálenosti od zdroje.
   b) Odstraní se dlouhé cesty (delší než do vrstvy obsahující stok) a také se odstraní zpětné hrany a hrany uvnitř vrstev.
   c) Nakonec se vytvoří fronta. Fronta zachycuje vrcholy v, které budou smazány. Z fronty postupně budeme mazat vrcholy pro které platí, že $deg^-(v) = 0$ nebo $deg^+(v) = 0$. Současně budeme mazat i hrany, které do těchto vrcholů vedou nebo z nich vycházejí. Pokud při odstraňování hran se některému vrcholu sníží stupeň na 0 bude přidán do fronty a následně bude smazán. Následně zlepšíme tok f podle blokujícího toku.

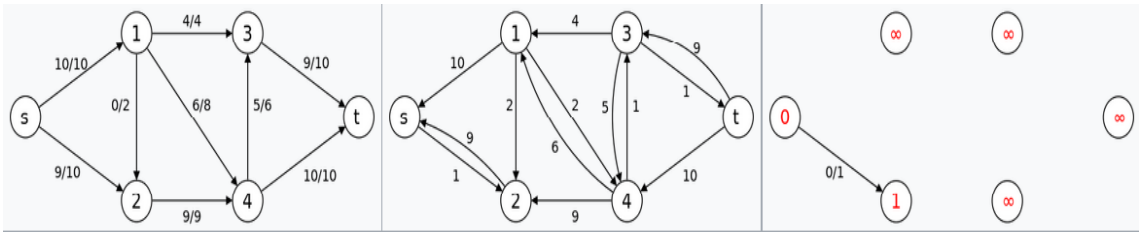
5.2.1 Příklad Dinicova / Edmonds Karpova algoritmu



Obrázek 8: Vytvoření reziduální sítě G_f, čisté sítě G_L a nalezení blokujícího toku [11]



Obrázek 9: Dočištění sítě a nalezení blokujícího toku [11]



Obrázek 10: Nalezení max. toku pomocí Dinicova / Edmonds Karpova algoritmu [11]

5.3 Metoda tří Indů

Indové Malhotra, Maheshawari a Pramodh - Kumar v roce 1978 objevili způsob jak zefektivnit Dinicův algoritmus pro hledání blokujícího toku v čisté síti. Předpokladem tohoto algoritmu je jednoduchá síť. Tím se považuje síť bez smyček a paralelních hran. Dále se neuvažují hrany vcházející do zdroje a vycházející ze stoku, jelikož tyto hrany nemůžou zvýšit maximální tok. V daném algoritmu si nejprve musíme spočítat rezervu vrcholu v, což je největší možný průtok přes daný vrchol. Průtok přes vrchol v budeme označovat $r(v)$ a vypočteme ho jako

$$r(v) := min\left\{r^+(v), r^-(v)\right\}, \tag{12}$$

kde
$r^+(v) = \sum_{uv \in E} r(uv)$ je součet rezerv všech hran vstupujících do vrcholu v a
$r^-(v) = \sum_{uv \in E} r(vu)$ je součet rezerv všech hran vystupujících z vrcholu v.

Jelikož hledáme blokující tok stačí uvažovat pouze rezervy po směru hrany. Rezervu po směru hrany vypočítáme opět jako rozdíl kapacity a toku na dané hraně. Pro nalezení blokujícího toku postupujeme následovně:

1. V prvním kroku opět přiřadíme k hranám nulový tok f.

2. Dále zjistíme rezervy všech hran u každého vrcholu. Rezervy se přepočítávají při každé změně toku.

3. Algoritmus se zastaví v momentě, kdy mají všechny vrcholy nulovou rezervu. V případě, že v síti existují vrcholy s nenulovou rezervou, vybereme vrchol v s nejmenší hodnotou $r(v)$.

4. Dále vrcholu v přiřadíme $p(v) = r(v)$, kde $p(v)$ představuje množství, které chceme přepravit ze zdroje do vrcholu v. Ostatním vrcholům bude přiřazena nulová hodnota. Postupně budeme po jednotlivých vrcholech ve vrstvách postupovat od vrcholu v ke zdroji z. Pro každý vrchol w budeme provádět následující operace dokud $p(w) > 0$:

   - Vybereme libovolnou hranu (u, w) a zvýšíme po ní tok o $\delta = min(r(u, w), p(w))$. Hodnota $p(w)$ se sníží o δ a hodnota $p(u)$ se zvýší o δ.
   - Pokud je hrana (u, w) nasycená odstraníme ji ze sítě a následně provedeme dočištění sítě stejným způsobem jako u Dinicova algoritmu,.

5. Nakonec stejným způsobem převedeme množství $r(v)$ z vrcholu v do spotřebiče s.

5.4 Škálovací algoritmus

Škálovací algoritmus funguje dobře pro sítě s celočíselnými kapacitami hran. U algoritmu je ze začátku zvolena hodnota δ, která je určena jako polovina maximální nalezené kapacity hrany vedoucí ze zdroje. Při výpočtu uvažujeme pouze hrany, které mají kapacitu větší nebo rovnu zvolenému δ. Jelikož se δ v každé iteraci zmenšuje na polovinu, postupně uvažujeme nové hrany.

Následně hledáme v každé iteraci zlepšující tok, který lze určit libovolným zvoleným algoritmem zmíněném výše, tedy například se může jednat o Dinicův algoritmus.

5.5 Goldbergův push - relabel algoritmus

Posledním algoritmem v této kapitole pro hledání maximálního toku je tzv. Goldbergův algoritmus (1988). Algoritmus používá dvě základní operace. Jednou z nich je protlačení toku po hraně *a druhou* operací je zvýšení výšky vrcholu. Zvednutí vrcholu, tedy zvýšení jeho výšky musí být provedeno tak, aby vrchol byl o 1 výš než nejnižší z následníků na rezervních hranách.

Dále si zavedeme pojem vlna, který je důležitým pojmem pro Goldbergův algoritmus. Funkce $f : E \to R_0^+$ je vlna v síti S, pokud splňuje tyto dvě podmínky

- $\forall e \in E : 0 \leq f(e) \leq c(e)$ (vlna nepřekročí kapacity hran),
- $\forall v \in V \setminus \{s, t\} : f(v) \geq 0$ (přebytek ve vrcholech je nezáporný).

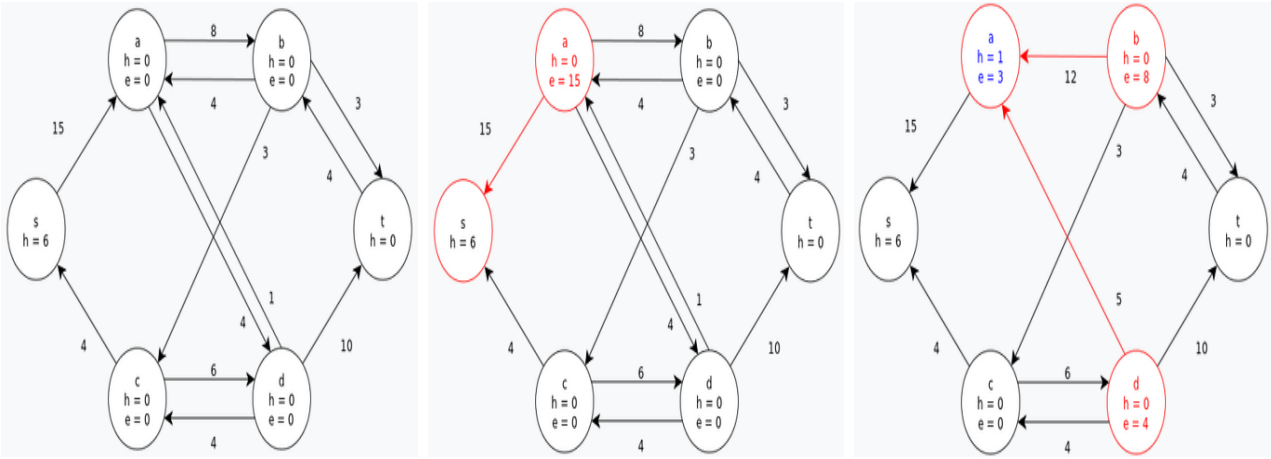
Po zavedení pojmu vlna platí tedy, že každý tok můžeme nazývat vlnou. Opačně to však platit nemusí. Vlna nemusí splňovat zákon zachování toku a tedy jsou ve vrcholech povoleny

přebytky. Pokud je přebytek ve vrcholech nezáporný a nenulový jedná se o aktivní vrchol. Zdroj a spotřebič nejsou nikdy aktivní.

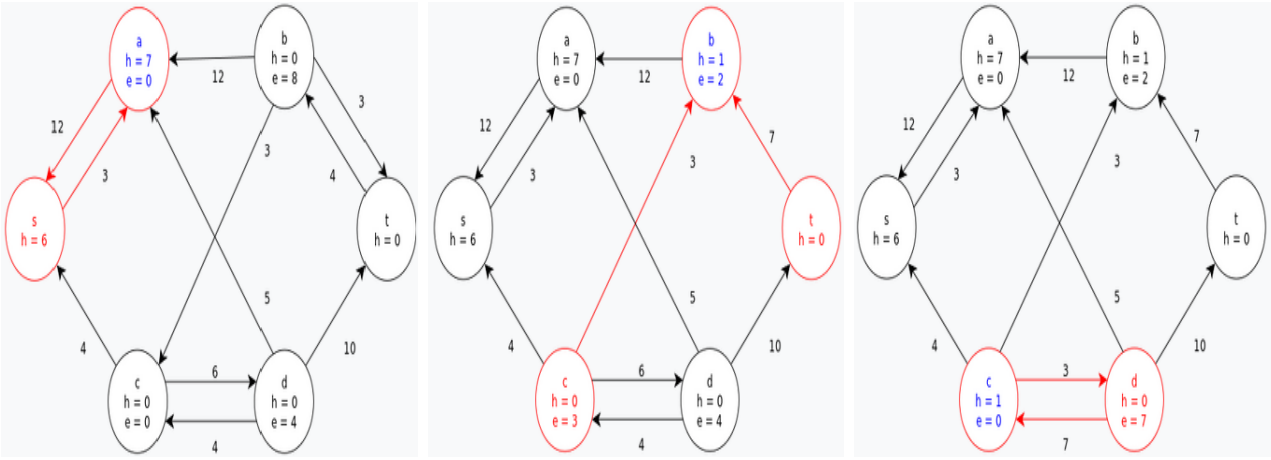
Postupně se budeme snažit, aby žádný z vrcholů nebyl aktivní, to znamená, že přebytek ve vrcholech bude nulový. Přebytky se snažíme převést do spotřebiče avšak pokud by vlna byla příliš velká dotlačíme přebytek z aktivních vrcholů zpět do zdroje. Vrcholům postupně přiřadíme výšky $h(v)$, které zavedeme kvůli zbytečnému přelévání přebytků tam a zpět. Na základě výšek potom budeme převádět přebytky a to z vyššího vrcholu do nižšího. Jestliže se vyskytne vrchol s přebytkem, ze kterého nevede žádná nenasycená hrana směrem dolů, zvedneme výšku tohoto vrcholu o jedna, dokud se nezvedne tak vysoko, aby z něj mohl přebytek odtéci. Algoritmus si nyní popíšeme v jednotlivých krocích:

1. Nastavíme počáteční výšky $h(v)$, kde zdroj bude ve výšce n a ostatní budou ve výšce 0.
2. Nasytíme všechny hrany vedoucí ze zdroje. Koncové uzly se stávají aktivními a jsou přidány do fronty.
3. Dokud není fronta aktivních uzlů prázdná budou provedeny tyto činnosti:
   - Odebereme uzel z fronty,
   - Pokud existuje nižší následník na rezervní hraně, protlačíme přebytek po této hraně,
   - V opačném případě zvedneme uzel o jedna,
   - Pokud je uzel stále aktivní, přidáme ho na konec fronty.

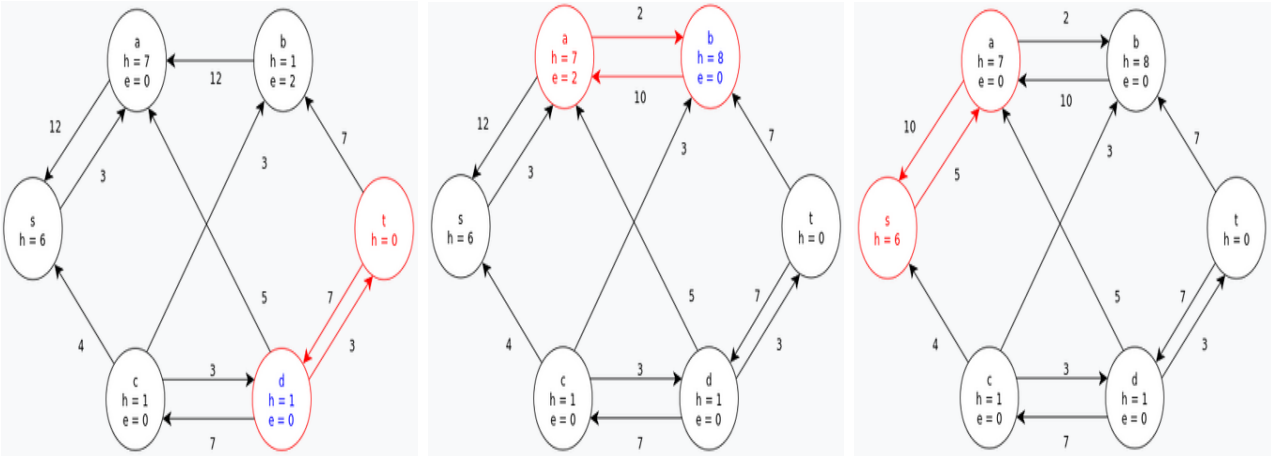
5.5.1 Příklad Goldbergova algoritmu



Obrázek 11: Nastavení výšek vrcholů, nasycení hran vedoucích ze zdroje a protlačení přebytku blíže ke spotřebiči [28]



Obrázek 12: Protlačování přebytku ke spotřebiči [28]



Obrázek 13: Nalezení maximálního toku podle Goldbergova algoritmu [28]

Vlastnosti, které se nemění při transformaci u Goldbergova algoritmu jsou následující:

1. Funkce f je v každém kroku vlna.
2. Výška $h(v)$ žádného vrcholu v nikdy neklesá.
3. $h(z) = n$ a $h(s) = 0$ po celou dobu běhu algoritmu.
4. $\forall$ vrchol v je $h(v) \leq 2n$.

5.6 Časová složitost algoritmů

Složitost algoritmu nám říká, jak je algoritmus rychlý. To znamená, kolik provede elementárních operací vzhledem k určité velikosti vstupních dat. K určení algoritmů se nejčastěji používá tzv. asymptotická složitost. Algoritmy se rozdělí do jednotlivých tříd složitostí. Pro každou třídu platí, že od určitého množství dat je algoritmus dané třídy vždy pomalejší než algoritmus třídy předchozí.

Jednotlivé třídy složitostí rozlišujeme na základě toho, že uvažujeme velké množství dat tj. $(n \to \infty)$. Potom neexistuje reálná konstanta taková, aby algoritmus z vyšší třídy byl rychlejší než ten z předchozí. Třídy složitostí nám udává následující stupnice (uvažujeme $k > 1$) [2]

$$1 \ll \log(n) \ll n \ll n\log(n) \ll n^k \ll k^n \ll n! \ll n^n$$

Složitost se zapisuje na základě tzv. Omikron notace jako $O(f(n))$. Rozlišujeme různé varianty složitosti:

1. **Horní odhad složitosti algoritmu** - udává složitost v nejhorším případě. Algoritmus probíhá asymptoticky stejně rychle nebo pomaleji než funkce $f(n)$. Označuje se jako $O(f(n))$.

2. **Dolní odhad složitosti algoritmu** - udává ideální složitost daného algoritmu (tedy nejrychlejší možné provedení), které je ale možné jen pro některé případy vstupních dat. Algoritmus probíhá asymptoticky stejně rychle nebo rychleji než funkce $f(n)$. Označuje se jako $\Omega(f(n))$

3. **Průměrný odhad složitosti algoritmu** - udává střední hodnotu složitosti při nějakém rozložení vstupních dat. Algoritmus probíhá asymptoticky stejně rychle jako funkce $f(n)$. Tento odhad nastává v případě, že algoritmus dosahuje horní složitosti jen „zřídka“. Označuje se jako $\Theta(f(n)$.

Nyní si uvedeme časovou složitost u algoritmů sloužících pro hledání maximálního toku v síti (viz. [9, str. 148-160]).

- Časová složitost u Ford - Fulkersonova algoritmu může být pro některé sítě a špatné volby zlepšujících cest příliš velká. Složitost algoritmu lze zlepšit tím, že budeme hledat zlepšující cesty na základě prohledávání do hloubky, přičemž bude vždy nalezena cesta, která má nejmenší počet hran ze všech dostupných cest. Maximální tok lze potom najít v čase $O(n\dot{m}^2)$, kde **n** je počet vrcholů v grafu a **m** je počet hran v grafu. Pokud by byly všechny kapacity rovny 1 složitost pro nalezení maximálního toku by byla $O(mn)$.

- Časová složitost u Dinicova / Edmonds - Karpova algoritmu se určí na základě rozdělení algoritmu na jednotlivé fáze (jednotlivé průchody vnějšího cyklu). Mazání hran s nulovou rezervou, sestrojení sítě rezerv, hledání vylepšující cesty a konečné zlepšování toku běží v čase $O(m)$.

  Dále veškeré dočišťování sítě během hledání blokujícího toku probíhá v čase $O(m)$. Každý vrchol i hrana jsou smazány nejvýše jednou za fázi, což znamená, že běží *v konstantním* čase.

  Hledání blokujícího toku projde nejvýše m cest, jelikož pokaždé bude odstraněna alespoň jedna hrana ze sítě. Nalezení cesty metodou trvá $O(n)$, tedy celkově je to $O(nm)$. Celá jedna fáze tedy běží v čase $O(nm)$.

  Jelikož každá cesta obsahuje nejvýše n vrcholů, tak proběhne nejvýše n fází. Celková časová složitost je tedy $O(mn^2)$. Pokud bychom vyjadřovali složitost v závislosti pouze na vrcholech byla by $O(n^4)$, jelikož hran grafu je nejvýše n^2.

- Algoritmus Metody tří Indů je modifikace Dinicova algoritmu a dokáže zpracovat síť *o* $\boldsymbol{n}$ vrcholech v čase $O(n^2)$. Místo hledání cesty se pro každý vrchol spočítá, jaké mají rezervy vstupní hrany a výstupní hrany jednotlivých vrcholů. Potom vrcholem s nejmenším minimem těchto hodnot se tok zvýší. Asymptotická složitost celého programu je potom $O(n^3)$. Této složitosti bylo dosaženo, jelikož se především zaměřujeme na vrcholy, nikoliv na hrany jako u Dinicova algoritmu.

- Asymptotická složitost u škálovacího algoritmu závisí logaritmicky na nejvyšší hodnotě kapacity hrany. Složitost je dále určena vzhledem k samotnému zvolenému algoritmu. Kdybychom uvažovali například Dinicův algoritmus byla by složitost rovna $O(m^2 \cdot \log_2 C)$, kde C vyjadřuje nejvyšší kapacitu z jednotlivých hran. Parametr δ nám udává, že v každé iteraci lze nalézt maximálně m zlepšujících cest. Nalezení cesty běží v čase $O(m)$.

- Asymptotická složitost u Goldbergova algoritmu je opět určena na základě složitostí jednotlivých operací. Složitost inicializace algoritmu je $O(m)$. Dále algoritmus provede nejvýše $2n^2$ zvednutí, což trvá $O(n)$. Provede nejvýše nm nasycených převedení a nejvýše n^2m nenasycených převedení. Převedení přebytku běží v konstantním čase $O(1)$. Celkově tedy najde algoritmus maximální tok v čase $O(n^2m)$.

Přehled časových složitostí u jednotlivých algoritmů je shrnut v následující tabulce.

Algoritmus	Časová složitost
Ford - Fulkerson	$O(nm^2)$
Dinic - Edmons Karp	$O(n^2m)$
Metoda tří Indů	$O(n^3)$
Škálovací	$O(m \cdot \log_2 C)$
Goldberg	$O(n^2m)$

Tabulka 1: Přehled časových složitostí

5.7 Cirkulace

5.7.1 Definice. Nechť je dán orientovaný graf $G = (V, E)$ a dále dolní a horní omezení l, c. Potom funkci $f : E \to \mathbb{R}$ nazveme cirkulací, pokud v každém vrcholu splňuje Kirchhoffův zákon, jinými slovy pokud platí, že

$$\sum_{uv \in V} f(uv) - \sum_{vu \in V} f(vu) = 0 \tag{13}$$

Přípustnou cirkulací nazveme cirkulaci pro kterou platí, že $l(e) \leq f(e) \leq c(e)$ pro každou hranou $e \in E$. Jestliže v síti G existuje tok ze zdroje do spotřebiče, který je přípustný, potom lze z tohoto toku vytvořit přípustnou cirkulaci. Přípustnou cirkulaci lze vytvořit přidáním hrany (t, s), přičemž necháme touto hranou téci tok rovný velikosti $|f|$. Pro hledání maximálního toku se tedy stačí omezit na úlohu hledání přípustné cirkulace, jelikož jsou úlohy ekvivalentní.

5.7.2 Věta. [10] Přípustná cirkulace v síti s dolním a horním omezením existuje jen tehdy, pokud pro každou množinu vrcholů V platí, že tok, který do ní musí vtéci (dolní omezení hrany), může z množiny V odtéci (horní omezení hrany).

Přípustnou cirkulaci lze najít pomocí algoritmu **Out of Kilter**, který si však nebudeme *v práci* uvádět. Jeho postup lze najít v odkazované literatuře [1, str. 326-328].

6 Algoritmy pro určení toku s minimální cenou

V následující kapitole si uvedeme některé základní algoritmy pro nalezení optimálního toku *s minimální* cenou [21]. V první části si uvedeme základní věty, definice a podmínky optimality řešení, které jsou nezbytné pro pochopení problematiky (viz. [1, str. 306-312], [22, str. 37-56] a [19, str. 190-206]). V odkazech lze nalézt také důkazy k jednotlivým tvrzením.

6.1 Předpoklad. V této kapitole budeme používat graf $G = (V, E, \alpha, \omega)$ pro označení konečného grafu, kde V označuje vrcholy grafu G, E označuje hrany grafu G, α označuje počáteční vrchol hrany a ω označuje koncový vrchol hrany. Dále budeme používat l a c, což jsou reálné funkce, které přiřazují dolní a horní kapacitu hrany v grafu G, kde platí, že $0 \leq l(e) \leq c(e)\ \forall e \in E$. Povolíme také případ, že $c(e) = +\infty$. To znamená, že některé hrany $e \in E$ mohou mít neomezenou kapacitu.

Pokud $b : V \to \mathbb{R}$ je libovolná funkce, potom $f : E \to \mathbb{R}$ nazýváme b-tok, jestliže platí, že $f(v) = b(v)\ \forall v \in V$, kde $f(v)$ označuje přebytek ve vrcholu v. Ve speciálním případě, kdy $b(v) = 0\ \forall v \in V$, nazýváme b-tok cirkulace. Budeme používat funkci $b : V \to \mathbb{R}$, ve významu požadovaných přebytků vrcholů. Dále definujeme funkci $k : E \to \mathbb{R}$, která přidělí jednotkové ceny toku k hranám. Pro b-tok f je cena určena jako

$$k(f) := \sum_{e \in E} k(e) \cdot f(e). \tag{14}$$

Nákladovou funkci k rozšíříme na hrany reziduální sítě G_f tak, že přiřadíme $k(+e) := k(e)$ a $k(-e) := -k(e)$, kde $+e$ označuje dopřednou hranu a $-e$ označuje hranu zpětnou. Jestliže $h : E(G_f) \to \mathbb{R}$, definujeme cenu ve tvaru

$$k(h) := \sum_{e \in E(G_f)} k(e) \cdot h(e). \tag{15}$$

$E(G_f)$ označuje hrany v reziduální síti, což jsou hrany, které mají kladnou rezervní kapacitu, jak je již zmíněno výše v kapitole (4).

6.2 Definice. (Problém s minimální cenou toku)
Pokud budeme vycházet již z výše uvedeného předpokladu (6), potom problém s minimální cenou toku je najít b-tok f, takový který je přípustný vzhledem k dolní a horní kapacitě, to znamená, že $l(e) \leq f(e) \leq c(e)\ \forall e \in E$ a jehož cena $k(f)$ je minimální mezi všemi možnými b-toky.

6.1 Podmínky optimality

Pro následující podmínku optimality řešení se ukáže, že je užitečné zapisovat rozšíření toku podél cyklu C v grafu G_f jako součet f a cirkulace β_C. Cirkulaci na cyklu (nazýváno jako tok po cyklu) s hodnotou $\delta > 0$ definujeme ve tvaru

$$\beta(e) := \begin{cases} \delta, & e \in E(C) \\ 0, & jinak. \end{cases} \tag{16}$$

Nechť f je b-tok a β_C je cirkulace na cyklu v G_f s hodnotou $\delta > 0$, potom definujeme $f + \beta_C$ ve tvaru

$$(f + \beta_C)(e) := \begin{cases} f(e) + \delta, & pokud + e \in C \\ f(e) - \delta, & pokud - e \in C \\ f(e), & jinak. \end{cases} \tag{17}$$

Je zřejmé, že $f + \beta_C$ je opět b-tok v G a jeho cena je $k(f + \beta_C) = k(f) + k(\beta_C)$.

6.3 Věta. [22, str. 38] Pokud f a f' jsou oba přípustné b-toky s ohledem na dolní a horní kapacity l *a* c, potom f' může být napsán jako součet f a maximálně $2m$ cirkulací na cyklech $\beta_{C_1}, \cdots, \beta_{C_p}$.

Následně dostaneme, že $k(f') = k(f) + \sum_{i=1}^{p} k(\beta_{C_i})$. Tato věta nám dává známou podmínku optimality pro minimální cenu toků.

6.4 Věta (Podmínka optimalizace cyklu pro minimální cenu toků [22, str. 39]). Nechť je G stejný jako v předpokladu (6). b-tok f má minimální cenu pouze tehdy pokud reziduální síť G_f neobsahuje cyklus se zápornou délkou (s ohledem na k).

Podmínky negativního optimálního cyklu naznačují jeden jednoduchý algoritmický přístup pro řešení problému minimálního ceny toku, který nazýváme algoritmus zrušení cyklu. Algoritmus se ukončí, jakmile reziduální síť neobsahuje žádný cyklus se zápornou délkou. Postup algoritmu můžeme najít zde [1, str. 317-318].

6.5 Věta. [22, str. 39] Jestliže existuje b-tok s minimální cenou, potom můžeme říci, že existuje také optimální tok f' takový, že $l(e) \leq f(e) \leq (n + m)(C + B)\ \forall e \in E$, kde $C = max\{c(e) : e \in E \wedge c(e) < +\infty\}$ a $B = max\{|b(v)| : v \in V\}$.

Nyní uvedeme druhou podmínku optimality. Nechť $G = (V, E, \alpha, \omega)$ je libovolný graf a dále $k : E \to \mathbb{R}$ je libovolný. Vzhledem k danému $p : V \to \mathbb{R}$ můžeme definovat také snížené ceny $k^p : E \to \mathbb{R}$ s ohledem na p ve tvaru

$$k^p(e) := k(e) + p(\alpha(e)) - p(\omega(e)). \tag{18}$$

Funkce $p : V \to \mathbb{R}$ se nazývá potenciál v G, právě tehdy pokud $p(\omega(e)) \leq p(\alpha(e)) + k(e)$ $\forall e \in E$. Potom je zřejmé, že p je potenciál pokud platí, že $k^p(e) \geq 0\ \forall e \in E$. Dále dokonce lze odvodit, že $k(C) = k^p(C) \geq 0$ pro všechny cykly C v G.

6.6 Věta. [22, str. 40] Nechť G je orientovaný graf a $k : E(G) \to \mathbb{R}$, potom v G existuje potenciál, pokud nemá G cyklus se zápornou délkou. Jestliže k je celočíselný, potenciál (pokud existuje) může být rovněž celočíselný.

Výše zmíněná věta nám dává dobrou druhou podmínku optimality pro minimální cenu toků.

6.7 Věta (Podmínka optimality snížené ceny pro minimální cenu toků [22, str. 40]). Nechť uvažujeme graf G, který je stejný jako v předpokladu (6). b-tok f má minimální cenu pouze

pokud existuje potenciál v G_f. To znamená, že existuje $p : V \to \mathbb{R}$, takové že $p(v) \leq k(e) + p(u)$ $\forall e \in G_f$, kde $u = \alpha(e)$ a $v = \omega(e)$.

6.8 Předpoklad. Po zbytek této kapitoly budeme vycházet z následujících tvrzení:

1. Existuje b-tok, který je přípustný vzhledem k dolním a horním kapacitám, zejména máme $\sum_{v \in V} b(v) = 0$.
2. $\forall u, v \in V$, přičemž $u \neq v$ existuje cesta z u do v, která se skládá jen z hran nekonečné kapacity $+\infty$ (tedy tato cesta existuje i v jakékoli reziduální síti G_f).
3. Budeme mít $l(e) = 0$ a $k(e) \geq 0$ $\forall e \in E$.

6.2 Postupný algoritmus nejkratší cesty

V této kapitole se budeme zabývat algoritmem nejkratší cesty k vyřešení problému toku *s minimální* cenou (viz. [22, str. 42], [1, str. 320-323] a [19, str. 199]). Jelikož algoritmus není časově polynomiální, budeme jej používat jako stavební blok pro odvození efektivnějších, tj. časově polynomiálních algoritmů v dalších částech.

6.9 Definice. (Pseudotok)
Nechť máme opět graf G, který je stejný jako v předpokladu (6) a $l \equiv 0$. Pseudotok v grafu G je funkce $f : E \to \mathbb{R}$, taková že $0 \leq f(e) \leq c(e)$ $\forall e \in E$, tj. nemusí splňovat zákon zachování toku. Pro pseudotok definujeme nerovnováhu (imbalanci) vrcholu $v \in V$ jako

$$imbal_f(v) = excess_f(v) - b(v), \tag{19}$$

kde $excess_f(v)$ označuje přebytek ve vrcholu v.

Pokud je $imbal_f(v) > 0$, nazýváme v jako vrchol s **přebytkem**. V případě, že je $imbal_f(v) < 0$, nazýváme v vrchol s **deficitem** a pokud má vrchol nulovou nerovnováhu nazývá se jako **vyvážený vrchol**.

Jednotlivé kroky algoritmu
Vstup: Orientovaný graf $G = (V, E, \alpha, \omega)$ s kapacitami $l \equiv 0$, $c : E \to \mathbb{R}^+$, požadovaný přebytek $b : V \to \mathbb{R}$ a ceny $k : E \to \mathbb{R}^+$.

1. Nastavíme $f(e) := 0$ $\forall e \in E$ a $p(v) := 0$ $\forall v \in V$.
2. Nastavíme $imbal_f(v) := -b(v)$ $\forall v \in V$.
3. Vypočteme množiny vrcholů s přebytkem a deficitem:

$$S_f := \{v \in V : imbal_f(v) > 0\}$$

$$D_f := \{v \in V : imbal_f(v) < 0\}$$

4. Dokud $S_f \neq \emptyset$ proveď kroky 5 - 12.

5. Vybereme $s \in S_f$ a $t \in D_f$.
6. Vypočteme vzdálenost $d(v) := dist_{k^p}(s, v, G_f)$ z vrcholu s do všech ostatních vrcholů v G_f vzhledem ke snížení cen k^p.
7. Nechť P je nejkratší cesta z s do t.
8. Nechť $\triangle := min\{c_f(e) : e \in P\}$.
9. Aktualizujeme $p := p + d$
10. $\epsilon := \{imbal_f(s), -imbal_f(t), \triangle\}$.
11. Zvýšíme f podél cesty P o ϵ jednotek.
12. Aktualizujeme f, S_f, D_f a G_f.
13. Ukončení kroku 4.
14. Algoritmus vrátí hodnotu f.

Příklad řešený pomocí algoritmu nejkratší cesty je uveden zde [1, str. 322]. Následující tvrzení (lemma) nám stanoví základy pro správnost algoritmu nejkratší cesty.

6.10 Lemma. [22, str. 43] Nechť f je pseudotok a $s \in V$. Nechť $G' \sqsubseteq G_f$ získáme odstraněním nějakých hran v G_f, takových že v G' jsou stále všechny vrcholy dosažitelné z vrcholu s. Předpokládáme, že $k^p(e) \geq 0$ $\forall e \in G'$ a označíme $d(v) := dist_{k^p}(s, v, G')$ pro $v \in V$. Potom následující tvrzení jsou pravdivé:

- Pro $p' := p + d$ máme také, že $k^{p'}(e) \geq 0$ $\forall e \in G'$.
- Jestliže e je hrana v G' na nejkratší cestě z s do t, potom $k^{p+d}(e) = 0$.

6.11 Důsledek. [22, str. 44] Důsledkem je, že nechť f je pseudotok, který splňuje podmínky optimality snížené ceny a f' je pseudotok získaný z f posláním toku z vrcholu s do nějakého jiného vrcholu t podél nejkratší cesty P v grafu G_f (s ohledem na snížené ceny k^p), potom f' také splňuje podmínky optimality snížené ceny.

6.12 Věta. [22, str. 44] Pokud budeme předpokládat, že všechna data jsou celočíselná, potom algoritmus nejkratší cesty končí po nejvíce nB iterací, kde $B = max\{|b(v)| : v \in V\}$. Pseudotok po ukončení je minimální cena b-toku. Celková doba chodu algoritmu je $O(nB(m+nlogn))$.

Pozorujeme, že ve skutečnosti není nutné inicializovat algoritmus nejkratší cesty průtokem $f \equiv 0$ a potenciálem $p \equiv 0$. Počet iterací algoritmu je potom omezen $\sum_{v \in V : imbal_f(v) > 0} imbal_f(v)$.

6.3 Postupný škálovací algoritmus

Tento algoritmus byl vůbec první časově polynomiální algoritmus pro problém minimální ceny toku (1972) [13]. Prezentace postupného škálovacího algoritmu v této části a algoritmu měřítka *a zmenšování* (scale-and-shrink) v následující části bude provedena pro přepravní problém (viz. [1, str. 360-363], [22, str. 44], [29, str. 121] a [19, str. 201]). To znamená, že budeme předpokládat $l(e) = 0$, $c(e) = +\infty$ a $k(e) \geq 0$ $\forall e \in E$. Tato prezentace dále vyžaduje ještě jeden předpoklad pro řešení přepravního problému. Vyžadujeme existenci speciálního vrcholu $z \in V$, takového že $b(z) = 0$, $(v, z) \in E$ s $k(v, z) = 0$ a $(z, v) \notin E$ $\forall v \in V \setminus \{z\}$. Tento předpoklad pouze zjednodušuje prezentaci algoritmu.

Instance (G, k, b) přepravního problému je dána grafem G, nezápornými ceny k a požadovanou nerovnováhou b. Postupný škálovací algoritmus byl vůbec první algoritmus pro problém toku s minimální cenou. Rozšířit (škálovat) instanci (G, k, b) přepravního problému pomocí faktoru $\triangle > 0$ znamená následující:

- Nahradíme $b(v)$ za $b'(v) = \lfloor b(v)/\triangle \rfloor$ $\forall v \neq z$.
- Nastaví se $b'(z) = -\sum_{v \neq z} b'(v)$.

6.13 Lemma. [22, str. 45] Nechť (G, k, b) je instance přepravního problému a (G, k, b') je získaná škálováním pomocí faktoru $\triangle$. Jestliže (G, k, b) má optimální řešení, potom (G, k, b') má také optimální řešení.

Škálovací algoritmus řeší instanci $I = (G, k, b)$ přepravního problému vyřešením řad stupňovaných instancí $I_L, I_{L-1}, \cdots, I_0$. Instance I_j je instance I škálovaná podle 2^j.

Jednotlivé kroky algoritmu pro řešení problému minimální ceny toku
Vstup: Orientovaný graf $G = (V, E, \alpha, \omega)$, nezáporné funkce ceny $k : E \to \mathbb{R}^+$, požadovaný vrchol nerovnováhy b.

1. Nechť $f \equiv 0$ a $p \equiv 0$
2. Nechť $L := 2^{\lfloor log_2 \overline{B} \rfloor}$, kde $\overline{B} := \sum_{v:b(v)>0} b(v)$.
3. Pro $j = L, L-1, \cdots, 0$ provedeme následující kroky:
4. Nechť I_j je instance získaná škálovaním I podle 2^j
5. Vyřešíme I_j podle algoritmu nejkratší cesty, který je inicializován dvojicí $(2f, p)$.
6. Aktualizace f je získaný optimální tok a p je odpovídající potenciál v G_f.
7. Ukončení kroku 3.

Postupný škálovací algoritmus řeší přepravní problém v čase $O(n \log \overline{B} S(n, m, nK))$, kde opět $\overline{B} := \sum_{v:b(v)>0} b(v)$ [22, str. 46].

6.4 Algoritmus měřítka a zmenšení

Ačkoliv doba běhu škálovacího algoritmu je polynomiální, není však silně polynomiální. To znamená, že počet aritmetických operací závisí i na velikosti čísel na vstupu. V této části uvedeme a analyzujeme algoritmus, který dosáhne silně polynomiální doby běhu (viz. [29, str. 123-126] a [22, str. 46]). Nyní odvodíme ještě další potřebnou podmínku optimality pro minimální cenu toků.

6.14 Věta (Doplňkové podmínky optimality uvolnění pro b-toky [22, str. 46]). Nechť G je graf stejný jako v předpokladu (6). Potom b-tok má minimální cenu pouze tehdy, pokud existuje $p : V \to \mathbb{R}$ takové, že snížené ceny k^p na hranách G splňují následující vlastnosti:

$$k^p(e) \geq 0 \Rightarrow f(e) = 0 \tag{20a}$$

$$l(e) \leq f(e) \leq c(e) \Rightarrow k^p = 0 \tag{20b}$$

$$k^p(e) < 0 \Rightarrow f(e) = c(e). \tag{20c}$$

Prezentace algoritmu v této kapitole bude opět provedena pro přepravní problém, to znamená, že opět předpokládáme $l(e) = 0$, $c(e) = +\infty$ a $k(e) > 0$ $\forall e \in E$. Prezentace opět vyžaduje existenci speciálního vrcholu z, u kterého opět platí, že $b(z) = 0$, $(v, z) \in E$ s $k(v, z) = 0$ a $(z, v) \notin E$ $\forall v \in V \setminus \{z\}$. Zopakujeme přepravní problém jako úlohu lineárního programování společně s jeho duální formulací, která se projeví jako užitečná v pozdější analýze:

$$min \sum_{e \in E} k(e) f(e) \tag{21a}$$

$$\sum_{e \in \delta^-(v)} f(e) - \sum_{e \in \delta^+(v)} f(e) = b(v) \qquad \forall v \in V \tag{21b}$$

$$f(e) \geq 0 \qquad \forall e \in E \tag{21c}$$

Duální formulace:

$$max \sum_{v \in V} b(v) p(v) \tag{22a}$$

$$p(\omega(e)) - p(\alpha(e)) \leq k(e) \qquad \forall e \in E \tag{22b}$$

Doplňkové podmínky uvolnění lineárního programování pro výše uvedené dvojice z dualních lineárních programů jsou následující:

$$f(e) > 0 \Rightarrow p(\omega(e)) - p(\alpha(e)) = k(e) \tag{23a}$$

$$\Rightarrow p(\omega(e)) - p(\alpha(e)) < k(e) \Rightarrow f(e) = 0. \tag{23b}$$

6.15 Lemma. [22, str. 47] Nechť (G, k, b) je přepravní problém s $b = 0$, tedy $b(v) = 0$ $\forall v \in V$. Potom jakékoliv přípustné řešení pro duální lineární program (22) je optimální řešení.

Poznámka: Pro algoritmus bude zaveden nezbytný pojem **Tight arc** (těsná hrana).

6.16 Definice. (Těsná hrana)
Hrana $e_0 \in E$ je těsná, pokud existuje optimální řešení přepravního problému s $f(e_0) > 0$.

Předpokládáme, že $e_0 \in E$ je těsná hrana pro instanci I přepravního problému, kde $u = \alpha(e_0)$ a $v = \omega(e_0)$. Doplňkové podmínky optimality uvolnění z věty 6.14 říkají, že $k^p(r_0) = 0$ pro jakékoliv optimální řešení p pro duální problém. Lze napsat ve tvaru

$$p(v) = k(e_0) + p(u) \tag{24}$$

pro jakékoliv optimální duální řešení. Rovnice (24) říká, že můžeme odstranit proměnnou $p(v)$ z duálního lineárního programu, tím že jej nahradíme $k(e_0) + p(u)$.

6.17 Věta. [22, str. 49] Nechť $I = (G, k, b)$ je instancí přepravního problému a e_0 je těsná hrana s $\alpha(e_0) = u$ a $\omega(e_0) = v$. Nechť $\hat{p}$ je optimálním řešením pro duální úlohu I/e_0 získané zadaným e_0. Potom optimální řešení p pro I je dáno jako

$$p(i) = \hat{p}(i) \qquad pro\ i \neq v \tag{25a}$$

$$p(v) = \hat{p}(u) + k(e_0). \tag{25b}$$

Jednotlivé kroky algoritmu pro řešení problému minimální ceny toku
Vstup: Orientovaný graf $G = (V, E, \alpha, \omega)$, nezáporné funkce ceny $k : E \to \mathbb{R}^+$, požadovaný vrchol nerovnováhy b.

1. Nastavíme $j := 1$ a $I = (G, k, b)$
2. Dokud $b \neq 0$ provedeme kroky 3 a 4
3. Zavoláme metodu FIND-TIGHT-ARC (I_j), abychom našli těsnou hranu e_0 z I_j a nastavíme $I_{j+1} := I_j/e_0$.
4. $j := j + 1$
5. Ukončení kroku 2
6. Najdeme přípustné řešení p pro dualitu z I_j
7. Dokud $j > 1$ provedeme kroky 8 a 9
8. Rozšíříme p na optimální duální řešení z I_{j-1}
9. $j := j - 1$
10. Ukončení kroku 7
11. Nalezneme přípustný tok f z (G, k, b), takový že $k^p(e) = 0$, kdykoliv $f(e) > 0$.

Pro realizace věty, která určuje těsnou hranu potřebujeme návrh stromového řešení pro přepravní problém. Stromové řešení je funkce $f : E \to \mathbb{R}$, která splňuje rovnováhu omezení (21b) a taková, že existuje strom T v G s $f(e) = 0\ \forall e \notin T$. Strom označuje souvislý graf, který neobsahuje žádný cyklus. Strom T jednoznačně určuje své stromové řešení (pokud existuje).

6.18 Lemma. [22, str. 50] Pokud existuje pro přepravní problém přípustné řešení, potom

existuje také řešení, které je stromové. Jestliže existuje optimální řešení, potom také existuje optimální řešení, které je stromovým řešením.

Podpogram pro nalezení těsné hrany

1. Nalezneme přípustné stromové řešení f pro (G, k, b)
2. Škálovaním instance (G, k, b) podle $\triangle = \frac{1}{n(n-1)} max\{f(e) : e \in E\}$ získáme novou instanci (G, k, b') s:
   - $b'(v) = \lfloor b(v)/\triangle \rfloor \ \forall v \neq z$
   - $b'(z) = -\sum_{v \neq z} b'(v)$
3. Nalezneme optimální stromové řešení f' pro instanci (G, k, b')
4. Nalezneme e_0, takový že platí $f'(e_0) \geq n - 1$

6.19 Lemma. [22, str. 51] Nechť $e_0 \in E$ je libovolná hrana taková, že v kroku 4 algoritmu zaměřujícího se na nalezení těsné hrany máme $f'(e_0) \geq n - 1$. Potom lze říci, že e_0 je těsná hrana.

6.20 Věta. [22, str. 53] Algoritmus měřítka a zmenšování (Scale-and-Shrink) nalezne optimální řešení pro přepravní problém v čase $O(n^2 \log nS(n, m, nK))$, kde $S(n, m, nK)$ je doba potřebná pro výpočet nejkratší vzdálenosti cesty z vrcholu v grafu G s **n** - vrcholy, **m** - hrany a nezápornými celočíselnými délkami v $\{0, \ldots, nK\}$.

7 Toky v dynamické podobě

Dynamické toky popisují strukturovanou síť a rozhodovací problémy v průběhu času, na které se v této kapitole zaměříme (viz. [20], [4], [22, str. 55-59] a [19, str. 207-208]). Tento důležitý model optimalizace sítě se objevuje například ve výrobních distribučních systémech, ekonomickém plánování, energetických systémech, komunikačních systémech, systémech manipulace s materiálem, dopravních systémech, železničních systémech, evakuačních systémech budov i v mnoha dalších oblastech. V této kapitole budeme předpokládat graf $G = (V, E)$, který znázorňuje souvislý graf bez násobných hran. Graf obsahuje dva vzájemně různé vrcholy $s, t \in V$ a $c : E \to \mathbb{R}^+$ je kapacitní funkce na hranách *v grafu* G. V dynamické podobě hrany *v* G navíc obsahují přepravní časy $\tau : E \to \mathbb{R}^+$. Pokud má hrana přepravní čas, potom říkáme, že tok, který je poslán z $\alpha(e)$ v nějakém čase t přes e dosáhne $\omega(e)$ v čase $t + \tau(e)$.

Nechť $T \in \mathbb{R}^+$. Potom funkce $f : E \times \mathbb{R} \to \mathbb{R}^+$ má časový horizont T, jestliže platí:

$$f(e, x) = 0 \ \forall e \in E \quad a \quad x \notin [0, T - \tau(e)], \tag{26}$$

kde $f(e, x)$ je průtoková rychlost v libovolném okamžiku $x \in \mathbb{R}$, kterou přiřazuje funkce f. Rovnice (26) udává, že f neodesílá tok před časem 0 a také, že posílá tok jen přes hranu e, pokud dosáhne $\omega(e)$ do času T. Pokud je f funkce s časovým horizontem T, potom dostáváme $\forall v \in V$:

$$\begin{aligned} excess_f(v, T) &= \sum_{e \in \delta^-(v)} \int_{-\infty}^{T} f(e, x - \tau(e))dx - \sum_{e \in \delta^+(v)} \int_{-\infty}^{T} f(e, x)dx \\ &= \sum_{e \in \delta^+(v)} \int_{0}^{T} f(e, x)dx - \sum_{e \in \delta^+(v)} \int_{0}^{T} f(e, x)dx, \end{aligned} \tag{27}$$

přičemž $f(v, T)$ nazýváme jako přebytek (excess) toku ve vrcholu v v čase $T \in \mathbb{R}^+$. Přebytek toku ve vrcholu v v čase θ je dán vztahem

$$excess_f(v, \theta) = \sum_{e \in \delta^-(v)} \int_{-\infty}^{\theta} f(e, x - \tau(e))dx - \sum_{e \in \delta^+(v)} \int_{-\infty}^{\theta} f(e, x)dx, \tag{28}$$

kde první suma udává přítok do vrcholu v do času θ a druhá suma udává odtok z vrcholu v do času θ.

7.1 Definice. (Dynamický tok s časovým horizontem T)
Nechť $s, t \in V$ jsou dva odlišné vrcholy v G, kde $s \neq v$. Dynamický (s, t)-tok s časovým horizontem T je funkce $f : E \times \mathbb{R} \to \mathbb{R}^+$ s časovým horizontem T, která splňuje následující omezení vážené-rovnováhy:

$$excess_f(v, \theta) \geq 0 \qquad \forall v \in V \setminus \{s, t\} \ \ a \ \ \forall \theta \in [0, T] \tag{29}$$

$$excess_f(v, T) = 0 \qquad \forall v \in V \setminus \{s, t\} \tag{30}$$

Pokud platí výraz (29) nazýváme f tokem bez čekání. V jiném případě nazýváme f tokem s čekáním. Jako obvykle nazýváme s a t zdrojem a spotřebičem dynamického toku. Hodnota dynamického toku sítě je dána jako

$$val(f) := excess_f(s, T), \tag{31}$$

to znamená, množství toku, které dosáhlo s v čase T. Můžeme říci, že f je přípustný vzhledem ke kapacitní funkci $c : E \to \mathbb{R}^+$, pokud $0 \leq f(e, \theta) \leq c(e)$, pro všechna $e \in E$ a pro všechna $\theta \in [0, T]$. Pokud f dynamický (s, t)-tok s časovým horizontem T, potom dostáváme, že

$$excess_f(s, T) + excess_f(t, T) = \sum_{v \in V} excess_f(v, T)$$

$$= \sum_{v \in V} \left(\sum_{e \in \delta^+(v)} \int_0^T f(e, x)dx - \sum_{e \in \delta^+(v)} \int_0^T f(e, x)dx \right) = 0. \tag{32}$$

Rovnice (32) nám říká, že množství toku, které opustí vrchol s se rovná množství toku, které dosáhne vrcholu t. Problém maximálního dynamického (s, t)-toku s časovým horizontem T je najít dynamický (s, t)-tok s časovým horizontem T maximální hodnoty.

7.1 Dynamické toky a řezy v průběhu času

Jako v případě statických toků, hrají řezy důležitou roli u dynamických variant řešení.

7.2 Definice. (Dynamický řez s časovým horizontem T)
Dynamický řez s časovým horizontem T je dán funkcí $X : [0, T) \to 2^V$, takovou že platí:

- $s \in X(\theta) \subseteq V \setminus \{t\}\ \forall\, \theta \in [0, T)$,
- $X(\theta_1) \subseteq X(\theta_2)$ pro $\theta_1 \leq \theta_2$.

Pro dynamický (s, t)-tok v grafu G platí, že každá jednotka toku musí v určitém čase překročit X na určité hraně v řezu X. Vezmeme například hranu $e = (u, v)$. V takovémto případě musí pro překročení řezu na e tok opustit vrchol u po čase ξ_u a dorazit do vrcholu v před časem ξ_v, tedy musí opustit vrchol u před časem $\xi_v - \tau(u, v)$. Hrana (u, v) je v řezu X během časového intervalu $[\xi_u, \xi_v - \tau(u, v))$. Výraz $\xi_v \in [0, T]\ \forall v \in V$ je pro dynamický (s, t)-tok definován ve tvaru

$$\xi_v := (\{\theta : v \in X(\theta')\} \cup \{T\}) \tag{33}$$

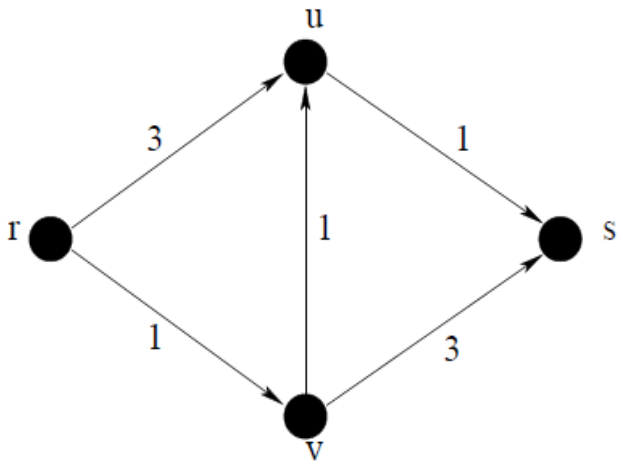
7.3 Definice. (Kapacita dynamického řezu)
Kapacita dynamického řezu X s časovým horizontem T se rovná množství toku, který je poslán přes hrany, přičemž hrany jsou v řezu.

$$c(X) := \sum_{(u,v) \in E} max\,\{0, \xi_v - \tau(u, v) - \xi_u\} \cdot c(u, v). \tag{34}$$

7.4 Lemma. [22, str. 57] Pokud f je dynamický (s, t)-tok s časovým horizontem T a pokud X je dynamický řez se stejným časovým horizontem T v grafu $G = (V, E)$, potom hodnota toku f je nejvýše kapacita dynamického řezu X.

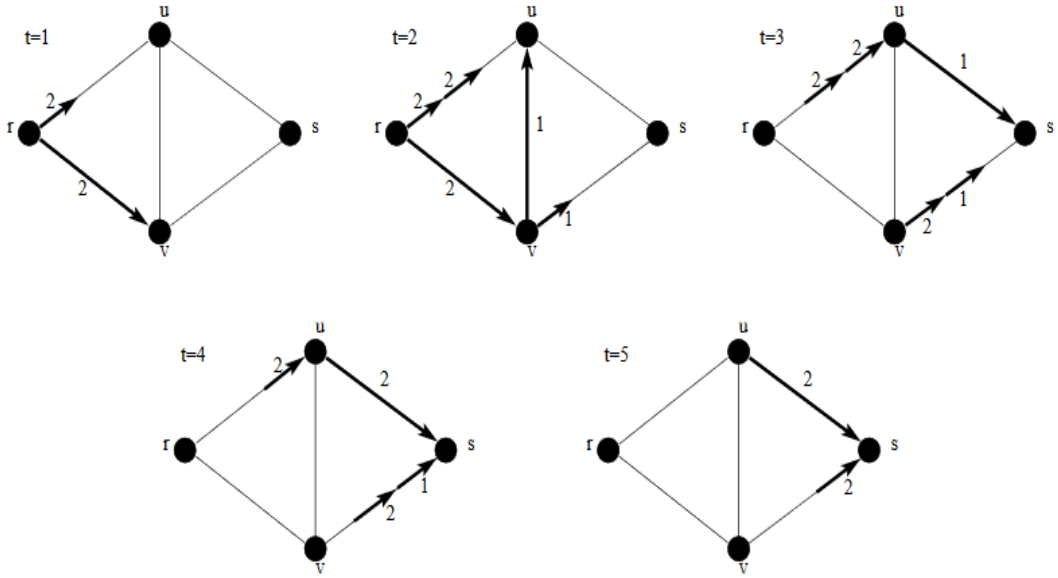
7.2 Příklad dynamického toku

Budeme uvažovat následující příklad dynamického toku, kde označení hran jsou přepravní časy a každá hrana má kapacitu 2. Na obrázku 14 je zobrazena síť pro řešení dynamického toku.



Obrázek 14: Dynamická síť
[20, str. 9]

Na následujícím obrázku 15 je zobrazen další postup pro hledání dynamického toku. Daný tok má propustnost 8 v čase $T = \{0, 1, \ldots, 5\}$. Grafy nám postupně poskytují případy *v jednotlivých* časech. Zadané hodnoty udávají množství toku, které je odesláno. Tučné části hran udávají část, kterou zachytí průtokové jednotky během jednotlivých časů.



Obrázek 15: Dynamický tok v čase 1 až 5
[20, str. 9]

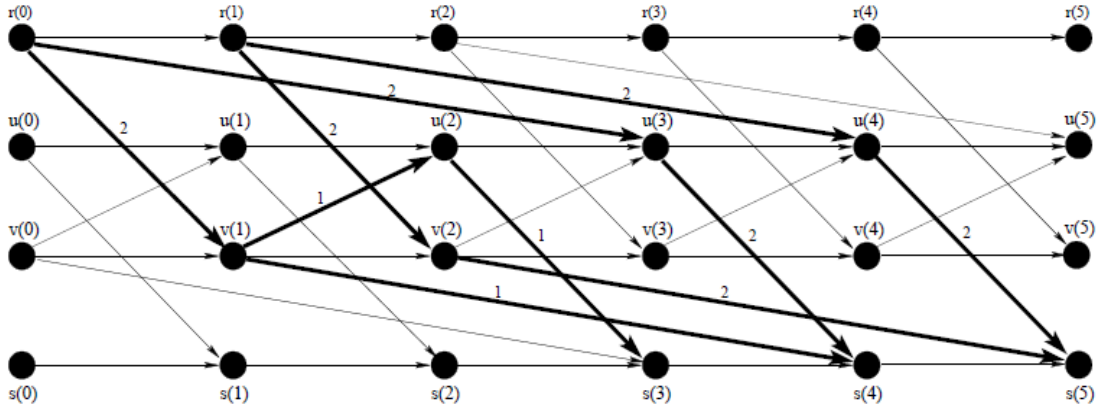
Mezi hlavní problémy toků, které lze řešit v dynamické síti patří maximální tok, nejrychlejší tok, minimální cena toku, nulový přepravní čas a složitost.

7.3 Techniky pro řešení problému dynamického toku

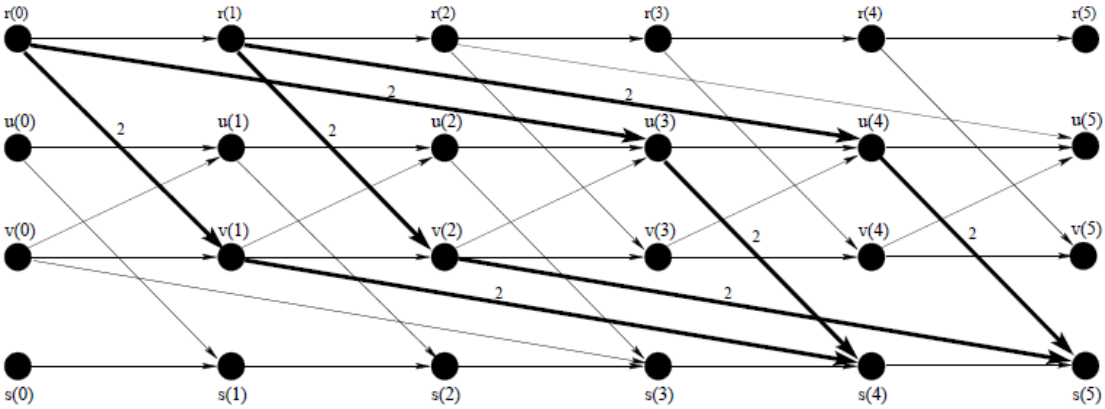
V této kapitole budeme popisovat metody, které slouží pro řešení problémů dynamického síťového toku.

7.3.1 Časově rozšířené grafy

První technikou pro řešení problému dynamického toku byly časově rozšířené grafy (time - expended graphs) [25]. Časově rozšířená verze dynamické sítě je statická síť, která obsahuje kopie vrcholů v jednotlivých krocích v určitém časovém horizontu $\{0, \ldots, T\}$ a hrany jsou překresleny mezi jednotlivými kopiemi a vyjadřují jejich přepravní časy. Na obrázku 16 je zobrazený tvar časově rozšířeného grafu z obrázku 14 v časovém horizontu $T = \{0, \ldots, 5\}$. Obrázek 17 potom zobrazuje maximální dynamický tok.



Obrázek 16: Časově rozšířená verze dynam. sítě z obrázku 14 a ekvivalent toku na obrázku 15 [20, str. 16]



Obrázek 17: Maximální dynamický tok v síti z obrázku 14 [20, str. 18]

7.5 Definice. (Časově rozšířená verze dynamické sítě G)
Je diagram G^T s vrcholy $v(t)$ $\forall v \in V$ a $t \in \{0, \cdots, T\}$. Dále pro každou hranu $(u, v) \in E$

s přepravním časem τ_{uv} a kapacitou c má graf G^T hrany $(u(t), v(t+\tau))$ s kapacitou c pro $t = 0, \cdots, T - \tau_{uv}$.

Tok na obrázku 14 odpovídá toku, který je zobrazen tučnými hrany na obrázku 15. Tok se dá transformovat na statický tok v G^T na problém s jedním zdrojem a spotřebičem zavedením právě společného zdroje a společného spotřebiče. Potom lze problém dynamického toku řešit nalezením statického toku v časově rozšířeném grafu.

7.3.2 Časově opakované toky

Časově opakované toky (Temporally Repeated flows) je velmi důležitý pojem pro nalezení maximálního dynamického toku.

7.6 Definice. (Časově opakované toky)
Pro statický tok f s rozkladem toku (viz. [7, str. 104] a [22, str. 10]) $\left\{\hat{f}_P : P \in P_1\right\}, \{\beta_C : C \in C_1\}$ definujeme časově opakovaný dynamický tok s časovým horizontem T, vyvozeným podle f, označeným f^T následovně: Pro $P \in P_1$ odešleme $val(\hat{f}_P)$ jednotek toku podél cesty P během časového intervalu, který je $[0, T - \tau(P)]$.

Platí, že $\hat{f} : E \to \mathbb{R}^+$ je statický (s,t)-tok v G. Dále jsme vycházeli z toho, že $\hat{f}$ lze rozvinout do toků na (s,t)-cestách a do cirkulací na cyklech. Označením P_1 jsou myšleny všechny (s,t)-cesty v G a C_1 udává všechny cykly v G. Nakonec výraz $\tau(P) := \sum_{e \in P} \tau(e)$ označuje celkový přepravní čas cesty P.

7.7 Věta. [22, str. 58] Pokud $\hat{f}$ je přípustný statický (s,t)-tok a f^T je časově opakovaný tok vyvozený podle $\hat{f}$, potom je f^T dynamický tok s časovým horizontem T a jeho hodnota je ve tvaru

$$val(f^T) = T\, val(\hat{f}) - \sum_{e \in E} \tau(e)\hat{f}(e). \tag{35}$$

Důkaz. Potřebujeme pouze dokázat rovnici o hodnotě toku. Tok f^T během časového intervalu $[0, T - \tau P]$ odesílá $val(\hat{f}_P)$ jednotek toku podél cesty $P \in P_1$. Tedy množství toku, které dosáhne spotřebič t do doby T je dáno:

$$\begin{aligned} val(f^T) &= \sum_{P \in P_1} val(\hat{f}_P)(T - \tau(P)) \\ &= T \sum_{P \in P_1} val(\hat{f}_P) - \sum_{P \in P_1} \sum_{e \in P} \tau(e) val(\hat{f}_P) \\ &= T val(\hat{f}) - \sum_{e \in E} \tau(e) \sum_{P \in P_1 : e \in P} val(\hat{f}_P) \\ &= T val(\hat{f}) - \sum_{e \in E} \tau(e)\hat{f}(e) \end{aligned}$$

□

7.8 Poznámka. Časově opakovaný tok f^T s maximální hodnotou $val(f^T)$ lze spočítat pomocí

metod pro výpočet problému minimální ceny toku v grafu $\overline{G}$, který je rozšířen o novou hranu e_{ts}.

Z rovnice (35) vyplývá, že hodnota f^T je nezávislá na rozložení. Pokud bychom uvažovali přepravní časy jako ceny $k(e) := \tau(e)\ \forall e \in E$ a přidali novou hranu e_{ts} z vrcholu t do vrcholu s, která by měla nekonečnou kapacitu a cenu $k(e_{ts}) = -T$, potom tok $\hat{f}$ minimalizuje objekt $T\,val(\hat{f}) - \sum_{e\in E} \tau(e)\hat{f}(e)$ pouze tehdy, pokud je $\hat{f}$ cirkulací v rozšířeném grafu $\overline{G}$. Tento problém je potom tedy problémem minimální ceny toku.

7.9 Věta. [22, str. 59] Pokud je f^T maximální opakovaný dynamický tok s časovým horizontem T, potom je také f^T maximální dynamický tok s časovým horizontem T.

Důkaz. Je uveden v [22, str. 59]. Na základě tohoto důkazu dostáváme následující větu:

7.10 Věta. Maximální hodnota dynamického (s,t)-toku s časovým horizontem T je rovna minimální kapacitě dynamického (s,t)-řezu s časovým horizontem T.

7.4 Časová složitost problémů v dynamické síti

V této kapitole si uvedeme výsledky časových složitostí problémů, které lze řešit v dynamické síti [20, str. 15]. Problém minimální ceny je NP-těžký. Třídy NP označují problémy, které jsou řešitelné na nedetermistickém Turingově stroji (na počítači) v polynomiálním čase. V každém kroku se výpočet rozvětví na **n** větví, ve kterých se hledá řešení současně. Problém pro hledání nejrychlejšího toku *v závislosti* na přepravních časech je silně NP-těžký. Nejznámější aproximace má poměr $(2+\epsilon)$. V sítích *s nulovým* přepravním časem lze nalézt řešení v polynomiálním čase. Složitost problémů toků *v průběhu* času ve srovnání se statickými tokovými problémy je shrnuta v následující tabulce [15].

Řešené problémy	Statické toky	Toky v průběhu času
(s, t)-tok	polynomiální	polynomiální
Přepravní problém	polynomiální	polynomiální
Problém minimální ceny toku	polynomiální	NP-těžký
Vícekomoditní toky	polynomiální	silně NP-těžký

Tabulka 2: Porovnání časových složitostí

Tabulka 2 porovnává výsledky časových složitostí. Problémy týkající se časově neměnných toků běží v polynomiálním čase. Jediné známé algoritmy polynomiálního času pro statické vícecomoditní výpočty toku vyžadují obecné lineární programovací techniky. Bylo dokázáno, že vícekomoditní tok v průběhu času s jednoduchými průtokovými cestami a bez ukládání toku v mezilehlých uzlech je silně NP-těžký. Nejznámějším výsledkem pro variantu silně *NP-těžkou s jednoduchými* průtokovými cestami a bez mezipaměti je 2-aproximační algoritmus pro problém s nejrychlejším průtokem vícekomoditních toků (viz. [23] a [24]).

Klinz a Woeginger ukázali, že problém výpočtu minimální ceny (s,t)-toku v průběhu času s předem stanovenou hodnotou a časovým horizontem je NP-těžký [6]. Hoppe a Tardos studovali nejrychlejší přepravní problém, který požaduje průtok v průběhu času a splňuje dané dodávky a požadavky na uzlech sítě v minimálním čase. Hoppe a Tardos dostali časově polynomiální algoritmus pro tento problém [5]. Nakonec Ford a Fulkerson dostali efektivní algoritmus, který řeší problém nejrychlejšího (s,t)-toku v polynomiálním čase [25].

V této kapitole jsme se setkali s pojmem vícekomoditní toky, to znamená, že lze v síti najednou přepravovat více subjektů. Problémy, které se týkají vícekomoditních toků jsou složitější *a obtížnější* pro řešitelnost. Vzhledem k rozsahu práce se těmito úlohami nebudeme dále zabývat.

8 Aplikace toků

Toky jsou nejvíce aplikovány v příkladech, které se týkají například vodovodního potrubí, silniční sítě nebo datové sítě. Pro řešení příkladu, kde například uvažujeme silniční síť je nutné znát dané body:

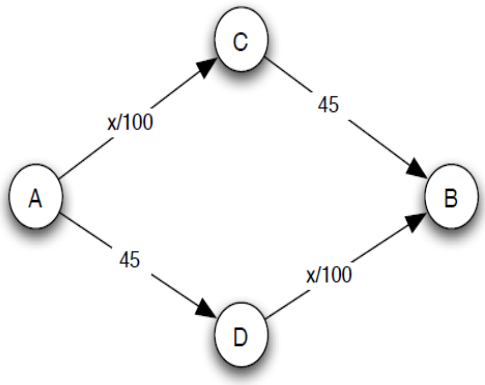
- Popis dopravní sítě - pro popis celé dopravní sítě se používá orientovaný graf,
- Vrchol (místo) ze kterého auta vyjíždějí,
- Vrchol (místo) do kterého auta dorazí,
- Kolik aut může danou hranou (silnicí) projet - používáme funkci kapacita a
- Kolik aut může danou hranou (silnicí) skutečně projet - používáme funkci tok.

U silniční sítě by tedy kapacita znázorňovala množství aut, které může danou silnicí projet za jednotku času (minutu). Úkolem by poté bylo najít maximální tok, tzn. kolik aut může projet ze zvoleného místa A do místa B. Pokud bychom navíc uvažovali na každé hraně cenu (cena za projetí silničním úsekem), řešili bychom problém toku s minimální cenou a cílem by bylo nalézt nejlevnější tok. Kdybychom na hranách uvažovali také přepravní časy, tj. čas potřebný k projetí daným úsekem, jednalo by se o problém hledání dynamického toku.

V následující podkapitole si představíme modelování sítě pomocí teorie her. Teorie her řeší případy, kdy výsledek rozhodnutí člověka nezávisí pouze na jeho volbě mezi několika možnostmi, ale také na volbách ostatních. Za zakladatele teorie her je považován John von Neumann, který ukázal možnost využití herně-teoretických modelů v oblasti modelování ekonomických rozhodovacích situací [32]. John von Neumann zde uvádí i nejznámější problém z teorie her tzv. Vězňovo dilema. Výběr trasy prostřednictvím dopravní sítě poukazuje právě na příklad, kdy výsledek každého rozhodovacího orgánu závisí na rozhodnutích ostatních. Rozhodování spočívá ve výběru trasy na základě času, který je potřebný k projetí daným úsekem.

8.1 Doprava v rovnováze

V této části kapitoly se budeme zabývat modelováním dopravní sítě pomocí teorie her, kdy se budeme snažit dostat z určitého místa A do určitého místa B. Reprezentujeme dopravní síť pomocí orientovaného grafu. Budeme se zabývat příkladem, kdy budeme vyvíjet model dopravní sítě a jeho reakci na dopravní zácpy (viz. [12] a [27]). V orientovaném grafu budeme uvažovat hrany jako dálnice a vrcholy nám budou reprezentovat body, do kterých se můžeme přes jednotlivé dálnice dostat. V grafu dále existují dva speciální vrcholy A a B, kde našim cílem je dostat ze z místa A do místa B, jak je již zmíněno výše. Každá hrana má také určený čas cesty. Čas závisí na množství dopravy, která se vyskytuje na jednotlivých hranách. Na obrázku 18 je nyní zobrazen orientovaný graf, který obsahuje již výše zmíněný postup.



Obrázek 18: Dopravní síť
[12, str. 230]

Označení na jednotlivých hranách nám udává dobu jízdy v minutách, pokud se na hranách vyskytuje x aut. V této dopravní síti jsou hrany $A \to D$ a $C \to B$ považovány za hrany, které nejsou citlivé na dopravní zácpy. Každá z těchto hran trvá 45 minut bez ohledu na počet vozidel, které cestují na nich. V případě, že se jedná o hrany $A \to C$ a $D \to B$ je z grafu vidět, že mají velký vliv na dopravní zácpy. Každá hrana trvá projet $x/100$ minut, pokud je na hraně $\boldsymbol{x}$ *aut*, které zde cestují.

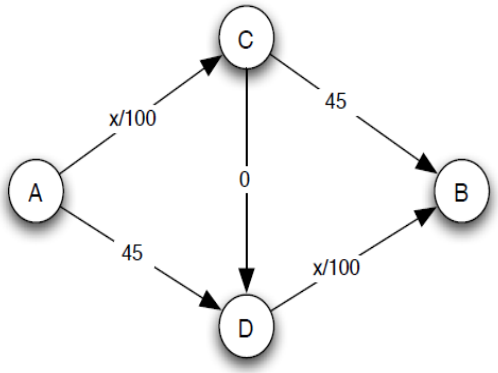
V dopravní síti, která je uvedena na obrázku 18 budeme uvažovat 4000 aut, který chtějí projet z bodu A do bodu B. Auta mají na výběr ze dvou cest a to buď přes vrchol C nebo přes vrchol D. Jestliže všechna auta pojedou přes jednu z výše uvedených cest celková doba jízdy na zvolené trase bude trvat 85 minut $(4000/100 + 45)$. Pokud však auta rozdělíme rovnoměrně mezi obě cesty tj. 2000 aut po každé cestě potom se celková doba jízdy zmenší na 65 minut $(2000/100 + 45)$.

Dopravní model je popsán jako hra, ve které hráči odpovídají řidičům a zvolené cesty od místa A do místa B jsou dané strategie hráčů. V tomto případě má každý hráč pouze dvě strategie. V modelu je uvažováno 4000 hráčů. Rovnováha nastává pouze v případě, že je $x = 2000$. *Z teorie* her se tento pojem označuje jako **Nashova rovnováha**, kdy platí, že hráč si svojí volbou nemůže zlepšit strategii, aniž by druhému hráči nezhoršil. Vyplývá to z faktu, že pokud uvažujeme, že na jedné z cest pojede $\mathbf{x}$ aut a na druhé cestě pojede $(4000 - x)$ aut, potom jestli není $x = 2000$ budou mít jednotlivé trasy odlišný čas cesty a řidiči (hráči) na pomalejší cestě budou motivováni ke změně volby přejít na rychlejší cestu. Proto jediná Nashova rovnováha nastává v případě $x = 2000$.

8.2 Braesův paradox

V této části popíšeme Braesův paradox, který ukazuje na to, že pokud je do sítě přidána nová kapacita, může tato kapacita ve skutečnosti zpomalit dopravní provoz.

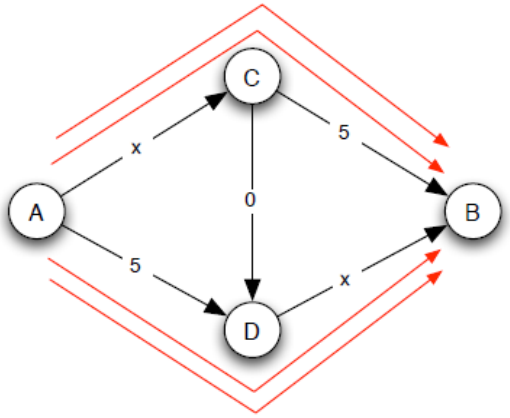
Budeme vycházet z obrázku 18. Změna v síti bude taková, že bude přidána nová hrana (silnice) z místa C do místa D. Pro jednoduchost budeme uvažovat, že doba jízdy mezi těmito místy je 0 a to bez ohledu počtu aut na cestě. Na obrázku 19 je nyní popsána nová dopravní síť *s přidanou* kapacitou.



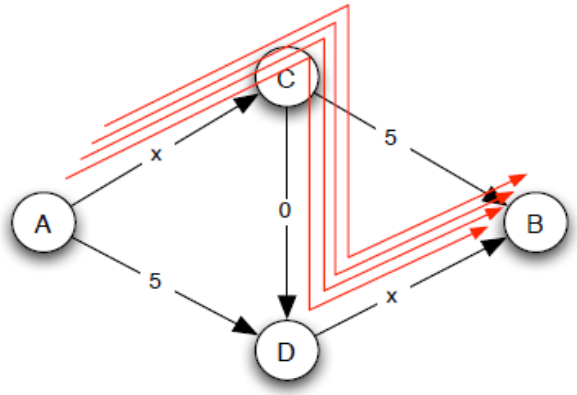
Obrázek 19: Rozšířená dopravní síť
[12, str. 232]

Na první pohled se zdá, že po přidání nové hrany by mělo být cestování z místa A do místa B lepší. Avšak v této nové dopravní síti existuje pouze jedna Nashova rovnováha, která však vede ke zhoršení doby cesty pro všechny řidiče. Rovnováha nastává v případě, že řidiči cestují přes místo C i přes místo D. To znamená, že doba cesty potom je $(4000/100 + 0 + 4000/100) = 80$ minut. Jiná cesta by v tomto případě trvala 85 minut a proto si všichni řidiči zvolí cestu $A \to C \to D \to B$. **Braesův paradox** poukazuje na to, že pokud je vytvořena nová rychlá silnice z C do D vtáhne do ní všechny řidiče. V nové síti potom neexistuje způsob, který by změnil chování řidičů, aby se vrátili k rovnoměrnému řešení, které by bylo pro každého lepší, jelikož doba cesty by trvala 65 minut.

V dopravním modelu vždy předpokládáme, že doba jízdy je buď pozitivní nebo nulová. Součet časů po zvolené cestě nazýváme jako sociální náklady. Předpokládáme síť, na které vznikne Braesův paradox a na které se vyskytují celkem čtyři řidiči. Minimálně možné náklady budeme považovat za sociálně optimální. Na obrázku 20 jsou nyní zobrazeny náklady, které jsou sociálně optimální a dosahují velikosti 28, jelikož každý řidič potřebuje 7 jednotek času k přemístění se z místa A do místa B. Na obrázku 21 je u téže dopravní sítě zobrazena Nashova rovnováha, která má však větší sociální náklady a to 32.



Obrázek 20: Sociální optimum pro 4 řidiče se sociálními náklady ve výši 28 [12, str. 235]



Obrázek 21: Nashova rovnováha pro 4 řidiče se sociálními náklady ve výši 32 [12, str. 235]

8.3 Optimalizace letového provozu

V této části si popíšeme, jak optimalizovat letový provoz [3]. Problém řízení letového provozu tak, aby byl zajištěn bezpečný a účinný proud letadel ve vzdušném prostoru se označuje jako ATFMP (Air Traffic Flow Management Problem). Postupy k řešení problému ATFMP jsou rozděleny do tří hlavních kategorií: Dlouhodobé, střednědobé a krátkodobé cíle. Řízení letového provozu (ATFM) je regulace letecké dopravy, taková aby se zabránilo překročení kapacity letišť nebo letového sektoru při manipulaci s dopravou. Cílem je přizpůsobit kapacitu systému letecké dopravy s požadavkem na to, aby bylo zajištěno, že letadlo může efektivně a bezpečně prolétat vzdušným prostorem. Dále je cílem zabránit přetížení a zpoždění. Počet letů odlétajících nebo přijíždějících z určitého letiště, jakož i počet přejezdů letadel v určitém odvětví vzdušného prostoru jsou funkce několika proměnných, které zahrnují počet dostupných přistávacích drah, kapacitu ATC (Air Traffic Control), omezení ve vzdušném prostoru a omezení týkající se toho, které letadlo může následovat letadlo dané třídy.

Jeden z hlavních problémů, s nimž se setkávají správci letového provozu, je problém

nalezení optimálních plánovacích strategií, které minimalizují náklady na zpoždění. Proto je potřeba nalézt dobré a optimální plánovací strategie ATFM, které nejen zmírní problémy *s přetížením*, ale také minimalizují náklady na zpoždění a současně uspokojují omezení kapacity letiště a vzdušného prostoru.

9 Závěr

V této práci jsme se zabývali úlohou hledání maximálního toku v síti. Z počátku jsme si zadefinovali několik základních pojmů, se kterými jsme během celé práce pracovali.

V kapitole 5 jsme se soustředili na časově neměnné toky. Dále jsme si uvedli algoritmy pro hledání maximálního toku v síti a to zejména Ford Fulkersonův, Dinicův / Edmonds Karpův, Goldbergův, Škálovací a nakonec algoritmus Metody tří Indů. Algoritmy byly postupně popsány v jednotlivých krocích a některé byly ilustrovány na příkladech pro lepší pochopení činnosti. *U algoritmů* byly dále určeny časové složitosti. Nakonec bylo zjištěno, že pro hledání maximálního toku se stačí omezit na úlohu hledání přípustné cirkulace, jelikož jsou úlohy ekvivalentní. Pro hledání přípustné cirkulace slouží algoritmus Out of Kilter, který byl pouze zmíněn v odkazované literatuře.

Hlavním cílem kapitoly 6 bylo popsat algoritmy pro určení toku s minimální cenou. Nejprve jsme uvedli algoritmus nejkratší cesty, který však není časově polynomiální a tedy nám posloužil jako stavební blok pro odvození efektivnějších (časově polynomiálních) algoritmů. Algoritmy byly popsány v jednotlivých krocích a prezentace těchto algoritmů byla provedena pro přepravní problém. Jednalo se o škálovací algoritmus a o algoritmus měřítka a zmenšení (scale and shrink). I v této kapitole byly určeny časové složitosti každého algoritmu.

V kapitole 7 jsme si definovali dynamické toky. V této části jsme se snažili najít maximální tok v dynamické síti, tedy v síti, která se může měnit v průběhu času. Byly uvedeny techniky pro řešení problému dynamického toku a to například časově rozšířené grafy. Dále byla úloha pro řešení hledání maximálního dynamického toku ilustrována na příkladu pro lepší pochopení. Uvedeny byly také základní problémy toků v dynamické síti, u kterých byla určena výpočetní složitost. Nakonec byly porovnány složitosti ve statické formulaci oproti dynamické.

Poslední část práce byla zaměřena na aplikace toků vybraných úloh v ekonomii. Zabývali jsme se modelováním dopravní sítě pomocí teorie her, kde jsme zkoumali, kdy nastane doprava *v rovnováze*. Uvedli jsme zde pojem Braesův paradox, který nám ukázal, že přidání nové kapacity do sítě může ve skutečnosti zpomalit provoz. Nakonec byla popsána optimalizace letového provozu a hlavní problém se kterým se můžeme u letového provozu setkat.

V práci byly zmíněny vícekomoditní toky, u kterých však nebyl zohledněn postup řešitelnosti vzhledem k rozsahu práce. Problémy, které se týkají řešitelnosti vícekomoditních toků bych se chtěl zabývat v budoucnosti a tedy tento návrh by mohl sloužit k případnému rozšíření práce.

Reference

[1] AHUJA, Ravindra K., Thomas L. MAGNANTI a James B. ORLIN. *Network Flows, Theory, Algorithms and Applications* [online]. New Jersey, 1993 [cit. 2017-05-24]. Dostupné z: http://cs.yazd.ac.ir/hasheminezhad/STSCS4R1.pdf

[2] Algoritmy.net. *Asymptotická složitost* [online]. [cit. 2017-05-24]. Dostupné z: http://www.algoritmy.net/article/102/Asymptoticka-slozitost

[3] ALOCHUKWU, Alex Somto. *On The Air Traffic Flow Management Rerouting Problem* [online]. 2016 [cit. 2017-05-25]. Dostupné z: https://www.wits.ac.za/media/wits-university/conferences/misgsa/documents/3.%20Air%20Traffic%20Flow%20Management.pdf

[4] ARONSON, Janine E. *A Survey of Dynamic Network Flows* [online]. The University of Georgia, USA, 1989 [cit. 2017-05-24]. Dostupné z: https://www.researchgate.net/profile/Janine_Aronson/publication/226470270_A_Survey_of_Dynamic_Network_Flows/links/54d26b660cf2b0c614696ecc.pdf

[5] B. Hoppe, E Tardos, *The quickest transshipment problem, Mathematics of Operations Research 25* (2000), pp. 36–62.

[6] B. Klinz, G.J. Woeginger, *Minimum-cost dynamic flows: The series–parallel case, Networks 43* (2004), pp. 153–162.

[7] BANG-JENSEN, Jorgen a Gregory GUTIN. *Digraphs, Theory, Algorithms and Applications* [online]. Springer, 2001 [cit. 2017-05-24]. Dostupné z: http://www.cs.rhul.ac.uk/books/dbook/main.pdf

[8] Books.fs.vsb.cz. *Teorie grafů*, Technická Univerzita Ostrava, 2006 [online]. [cit. 2017-05-24]. Dostupné z: http://books.fs.vsb.cz/SystAnal/texty/21.htm

[9] ČERNÝ, Jakub. *Základní grafové algoritmy* [online]. ČVUT v Praze, 2013 [cit. 2017-05-24]. Dostupné z: http://kam.mff.cuni.cz/~kuba/ka/ka.pdf

[10] DEMLOVÁ, Marie. *Teorie grafů* [online]. 2016 [cit. 2017-05-26]. Dostupné z: http://math.feld.cvut.cz/demlova/teaching/grafy/nezkont_pred/p-graf607.pdf

[11] *Dinic's Algorithm.* In: Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation, 2001 [cit. 2017-05-24]. Dostupné z: https://en.wikipedia.org/wiki/Dinic's_algorithm

[12] EASLEY, David a Jon KLEINBERG. *Networks, Crowds, and Markets: Reasoning About a Highly Connected World: Chapter 8* [online]. Cambridge University, 2010 [cit. 2017-05-25]. Dostupné z: http://www.cs.cornell.edu/home/kleinber/networks-book/

[13] J. Edmonds and R.M. Karp. *Theoretical improvents in algorithmic efficiency for network flow problem.* J. ACM, 1972, 248-264.

[14] FORD, Lester Randolph Jr. a Delbert Ray FULKERSON. *Flows in Networks* [online]. 1962 [cit. 2017-05-24]. Dostupné z: http://www.rand.org/pubs/reports/R375.html

[15] HALLA, A., S. HIPPLERB a M. SKUTELLA. *Multicommodity flows over time: Efficient algorithms and complexity: Theoretical Computer Science* [online]. 2007, s. 387-404 [cit. 2017-05-24]. Dostupné z: http://www.sciencedirect.com/science/article/pii/S0304397507001508#bb1

[16] HLINĚNÝ, Petr. *Základy teorie grafů* [online]. 2010 , s. 1-20 [cit. 2017-05-24]. Dostupné z: https://is.muni.cz/do/1499/el/estud/fi/js10/grafy/Grafy-text10.pdf

[17] Is.mendelu.cz. *Sítě a toky* [online]. [cit. 2017-05-24]. Dostupné z: https://is.mendelu.cz/eknihovna/opory/zobraz_cast.pl?cast=19941

[18] KAPLAN, Haim. *Advanced topics in algorithms* [online]. Spring, 2013 [cit. 2017-05-24]. Dostupné z: http://www.cs.tau.ac.il/~haimk/adv-alg-2013/

[19] KORTE, Bernhard a Jens VYGEN. *Combinatorial Optimization: Third Edition* [online]. 2006 [cit. 2017-05-24]. Dostupné z: http://cs.yazd.ac.ir/hasheminezhad/COF13R1.pdf

[20] KOTNYEK, Balasz. *An Annotated overview of dynamic network flows* [online]. RR-4936. INRIA, 2003 [cit. 2017-05-24]. Dostupné z: https://hal.inria.fr/inria-00071643/document

[21] KOVÁCS, Péter. *Minimum-cost flow algorithms: An experimental evaluation* [online]. Budapest, Hungary, 2013 [cit. 2017-05-26]. Dostupné z: https://www.cs.elte.hu/egres/tr/egres-13-04.pdf

[22] KRUMKE, Sven O. *Advanced Network Flows and Selfish Routing* [online]. University of Kaiserslautern, 2015 [cit. 2017-05-25]. Dostupné z: http://www.mathematik.uni-kl.de/fileadmin/AGs/opt/Lehre/SS16/flows/Notes-Complete1-9.pdf

[23] L. Fleischer, M. Skutella. *The quickest multicommodity flow problem*, in: W.J. Cook, A.S. Schulz (Eds.), *Integer Programming and Combinatorial Optimization*, in: Lecture Notes in Computer Science, vol. 2337, Springer, Berlin, 2002, pp. 36–53.

[24] L. Fleischer, M. Skutella. *Quickest flows over time*, SIAM Journal on Computing (2007) 36 (6), 1600–1630.

[25] L.R. Ford, D.R. Fulkerson. *Constructing maximal dynamic flows from static flows*, Operations Research 6 (1958), 419–433.

[26] MAREŠ, Martin. *Krajinou grafových algoritmů: Kapitola 1-2* [online]. Universita Karlova v Praze, 2006 [cit. 2017-05-24]. Dostupné z: http://mj.ucw.cz/vyuka/ga/

[27] NAGURNEY, Anna. *Network Economics: An Introduction* [online]. University of Massachusetts, 2002 [cit. 2017-05-25]. Dostupné z: https://supernet.isenberg.umass.edu/austria_lectures/fintrosl.pdf

[28] *Push - Relabel algorithm*. In: Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2017-05-24]. Dostupné z: https://en.wikipedia.org/wiki/Push%E2%80%93relabel_maximum_flow_algorithm

[29] SCHRIJVER, Alexander, William J. COOK, William H. CUNNINGHAM a William R. PULLEYBLANK. *Combinatorial Optimization* [online]. New York, 1998 [cit. 2017-05-24]. ISBN 0-471-55894-X. Dostupné z: https://www.google.cz/search?hl=cs&tbo=p&tbm=bks&q=isbn:1118031393

[30] ŠIŠMA, Pavel. *Vznik a vývoj teorie grafů* [online]. vol. 43 (1998), pp. 89-99 [cit. 2017-05-24]. Dostupné z: http://dml.cz/bitstream/handle/10338.dmlcz/137535/PokrokyMFA_43-1998-2_1.pdf

[31] Teiresias.muni.cz. *Adaptace Matematických Algoritmů* [online]. [cit. 2017-05-24]. Dostupné z: https://www.teiresias.muni.cz/amalg/www/cs/adaptation/fold-fulkerson

[32] von Neumann, J.; Morgenstern, O.: *Theory of Games and Economic Behavior*. Princeton University Press, Princeton, 1944.

Seznam obrázků

Seznam tabulek