Vysoké učení technické v Brně
Fakulta informačních technologií

Grafové algoritmy

Hledání párování maximální kardinality

Bc. Jiří Chromečka, xchrom12
Bc. Jan Vybíral, xvybir05

1 Úvod

Úkolem projektu bylo implementovat a porovnat dva různé algoritmy pro hledání párování maximální kardinality.

Zvolené grafy jsou nevážené, neorientované a bipartitní.

1.1 Hledání párování maximální kardinality

Uzly v bipartitním grafu se dělí do dvou disjunktních množin a hrany jsou podmnožinou jejich kartézského součinu.

Cílem je najít takovou množinu hran v tomto grafu, kdy žádné dvě hrany nemají společný uzel a množina má největší kardinalitu.

2 Hopcroft-Karp algoritmus

`Hopcroft-Karp` algoritmus využívá tzv. rozšiřující (augmenting) cesty, což jsou cesty, ve kterých se střídají hrany mimo párování s hranami z párování, počet hran je lichý a každá lichá hrana je mimo párování. Opakovaným nalezením množiny nejkratších rozšiřujících cest a symetrickým rozdílem s párováním získáme nové párování s větší kardinalitou. Algoritmus samotný cyklí mezi několika fázemi.

V první fázi pomocí prohledávání do šířky z volných uzlů první množiny se hledá nejkratší rozšiřující cesta a po jejím nalezení se všechny volné uzly z druhé množiny na této úrovní prohledávání využijí do další fáze.

Ve druhé fázi se prohledáváním do hloubky nad uzly z předchozí fáze hledají nevyužité uzly v předchozích úrovních prohledávání, dokud není nalezena rozšiřující cesta, která tedy zahrnuje uzel z předchozí fáze. Každá z nalezených rozšiřujících cest je použita v párování.

Algoritmus končí, když při prohledávání do šířky nenalezne žádnou nejkratší rozšiřující cestu.

2.1 Teoretická složitost

V grafu $G = (V, E)$, každá iterace fází trvá $\mathcal{O}(|E|)$, neboť jakmile je pomocí prohledávání do šířky nalezena nejkratší rozšiřující cesta, zpětné prohledávání do hloubky z této úrovně navíc uvažuje jen nenavštívené uzly a každá hrana je prozkoumána jen jednou.

Po $\sqrt{|V|}$ iteracích, nejkratší rozšiřující cesta v kolekci má délku alespoň $\sqrt{|V|}$ hran. Pokud toto platí pro každou cestu, může kolekce obsahovat nejvýše $\sqrt{|V|}$ cest. Tuto kolekci tvoří symetrický rozdíl optimálního párování a současného, kdy optimální bude mít nejvýše o $\sqrt{|V|}$ hran více než současné. Protože každá iterace zvýší počet hran párování alespoň o jednu, může proběhnout nejvýše $\sqrt{|V|}$ iterací, než algoritmus nalezne optimální párování, tedy $2\sqrt{|V|}$ celkově.

Teoretická složitost algoritmu je počet iterací * složitost jedné iterace, tedy $\mathcal{O}(|E| * \sqrt{|V|})$ v nejhorším případě.

3 Edmonds-Karp algoritmus

`Edmonds-Karp` algoritmus je verze `Ford-Fulkerson` algoritmu pro nalezení maximálního toku v síti. Na začátku se nejprve nastaví tok všemi hranami na 0. Poté se v cyklu opakovaně hledá cesta od zdroje ke spotřebiči v reziduální síti s pomocí prohledávání do šířky. V každé takto nalezené cestě se najde hrana s nejmenší reziduální kapacitou a o hodnotu této reziduální kapacity se zvýší tok danou cestou směrem od zdroje ke spotřebiči. Toto se provádí dokud lze v reziduální síti nalézt nějakou cestu od zdroje ke spotřebiči, na které je možné zvýšit tok (tzn. reziduální kapacita každé její hrany je alespoň 1).

Tento algoritmus lze využít i pro hledání maximálního párování v bipartitním gtafu. Nejprve se musí vstupní bipartitní graf (tvořený dvěmi množinami vrcholů A a B) převést na síť přidáním zdroje a spotřebiče. Všechny vrcholy z množiny A se potom napojí hranami na zdroj a tok těmito hranami se nastaví na 1 směrem od zdroje. Naopak všechny vrcholy z množiny B se napojí hranami na spotřebič a tok těmito hranami se nastaví na 1 směrem ke spotřebiči. Tok každou hranou z původního grafu se nastaví na 1 směrem z A do B. Kapacita všech hran se nastaví na 1.

Nad takto sestavenou sítí se poté spusí algoritmus `Edmonds-Karp` popsaný výše. Po jeho skončení se do výsledného maximálního párování přidá každá hrana, pro kterou platí, že její jeden vrchol je z množiny A, druhý její vrchol je z množiny B, a tok touto hranou je nenulový.

3.1 Teoretická složitost

Počáteční převod vstupního grafu na síť a její inicializace má složitost $\mathcal{O}(|E|)$.

Algoritmus prohledávání do šířky má složitost $\mathcal{O}(|V| + |E|)$. Nalezená cesta od zdroje ke spotřebiči musí mít vždy nanejvýš $|V| - 1$ hran, takže operace nalezení hrany v této cestě s nejnižší reziduální kapacitou a aktualizace toků a reziduálních kapacit na všech v cestě obsažených hranách má složitost $\mathcal{O}(|V|)$.

Jelikož všechny hrany mají kapacitu 1, zvýší se velikost toku v každé iteraci hlavního cyklu o 1. Protože počáteční velikost toku je 0, celkový počet iterací je roven velikosti maximálního toku. Je přitom zřejmé, že velikost maximálního toku je rovna kardinalitě maximálního párování, která je shora omezená počtem vrcholů. Takže celková složitost hlavního cyklu je $\mathcal{O}((|V| + |E|) * |V|)$. Ale jelikož graf je souvislý, tak platí, že $|E| \geq |V| - 1$, a tudíž celková složitost hlavního cyklu je $\mathcal{O}(|E| * |V|)$.

Je zřejmé, že závěrečné přidání všech hran s nenulovým tokem do maximálního párování má složitost $\mathcal{O}(|E|)$, celková složitost celého algoritmu `Edmonds-Karp` pro nalezení maximálního párování v bipartitním grafu je tedy $\mathcal{O}(|E|) * |V|)$.

4 Testování

Pro potřeby experimentálního testování byl napsán jednoduchý generátor bipartitních grafů se zvoleným počtem obou množin uzlů a hran, integrovaný v programu. Experimentální testování proběhlo na serveru `merlin.fit.vutbr.cz`. Pro graf $G = (B \cup G, E)$ byly voleny různé hodnoty počtu uzlů z první množiny B, počtu uzlů z druhé množiny G a počtu hran z množiny E. U toho byla změřena doba trvání t samotného algoritmu bez okolního kódu.

4.1 Hopcroft-Karp algoritmus

$\|B\|$	$\|G\|$	$\|E\|$	$t\,[sec]$
1000	10	10	0.01
1000	10	100	0.01
1000	10	1 000	0.02
1000	10	10 000	0.05
1000	100	10	0.01
1000	100	100	0.01
1000	100	1 000	0.02
1000	100	10 000	0.06
1000	100	100 000	0.48
1000	1000	10	0.02
1000	1000	100	0.03
1000	1000	1 000	0.04
1000	1000	10 000	0.13
1000	1000	100 000	0.54

Tabulka 1: Tabulka experimentů pro Hopcroft-Karp algoritmus

Experimentální testování prokázalo, že složitost algoritmu `Hopcroft-Karp` přibližně odpovídá $\mathcal{O}(|E| * \sqrt{|V|})$. Z provedených experimentů vyplývá, že je složitost algoritmu výrazně závislá na počtu hran grafu a přiměřeně závislá na počtu uzlů grafu. Algoritmus podává velmi dobré výsledky pro řídké grafy a o něco horší výsledky pro hustší grafy.

4.2 Edmonds-Karp algoritmus

$\|B\|$	$\|G\|$	$\|E\|$	$t\,[sec]$
500	500	1 000	1.86
900	100	1 000	0.81
1 000	1 000	1 000	4.71
1 900	100	1 000	1.59
1 500	1 500	1 000	8.03
2 900	100	1 000	2.47
2 000	2 000	1 000	12.34
3 900	100	1 000	3.23
2 500	2 500	1 000	15.89
4 900	100	1 000	4.26
3 000	3 000	1 000	21.73
5 900	100	1 000	5.08

$\|B\|$	$\|G\|$	$\|E\|$	$t\,[sec]$
1 000	1 000	500	2.76
1 000	1 000	1 000	4.75
1 000	1 000	1 500	6.45
1 000	1 000	2 000	7.96
1 000	1 000	2 500	9.55
1 000	1 000	3 000	11.38

Tabulka 2: Tabulky experimentů pro Edmonds-Karp algoritmus

Jak je vidět z tabulek výše, výsledky experimentů přibližně odpovídají výše uvedené teoretické časové složitosti algoritmu `Edmonds-Karp` $\mathcal{O}(|E|) * |V|)$. Čas potřebný pro jeho vykonání v naší implementaci roste přibližně lineárně s rostoucím počtem vrcholů (první tabulka) i hran (druhá tabulka). Z první tabulky je také patrné, že pro bipartitní grafy, které mají v jedné ze dvou množin vrcholů výrazně větší počet vrcholů než v druhé množině, skončí algoritmus při stejném celkovém počtu vrcholů i hran podstatně dříve. Toto zřejmě plyne z menšího počtu možných cest mezi zdrojem a spotřebičem v sítích vytvořených z takovýchto grafů.

5 Závěr

Implementovali jsme dva algoritmy pro výpočet maximálního párování v bipartitních grafech, `Edmonds-Karp` s teoretickou časovou složitostí $\mathcal{O}(|E|) * |V|)$ a `Hopcroft-Karp` s teoretickou časovou složitostí $\mathcal{O}(|E| * \sqrt{|V|})$. Nad naší implementací jsme provedli řadu experimentů, které prokázaly, že časová složitost naší implementace přibližně odpovídá uváděné teoretické časové složitosti pro tyto algoritmy. Podle očekávání byla doba nutná pro vykonání algoritmu `Edmonds-Karp` výrazně vyšší než doba nutná pro vykonání algoritmu `Hopcroft-Karp` na bipartitním grafu se stejným počtem uzlů a hran.

6 Ovládání programu

Program má celkem 6 přepínačů, které mohou být zadávány v libovolném pořadí:

$-h$: zobrazí nápovědu
$-c\ B\ G\ E$: vygeneruje bipartitní graf s počty uzlů B a G a počtem hran E
$-l\ F$: načte bipartitní graf ze souboru F
$-f\ M$: spočítá maximální párování metodou M (h pro `Hopcroft-Karp`, e pro `Edmonds-Karp`)

$-g$: vytiskne vstupní graf
$-m$: vytiskne maximální párování

Soubor se vstupním grafem musí být ve specifickém formátu: na každém řádku jsou jména uzlů oddělená středníky. První řádek obsahuje všechny uzly z první množiny, druhý řádek všechny uzly z druhé množiny a na ostatních řádcích jsou seznamy sousedů (vždy jako uzel následovaný svými sousedy). Viz. přiložený vzorový vstupní soubor `input.txt`.