Implementace Edmondsova algoritmu

Matěj Kocián*

22. března 2013

Edmondsův zahradní algoritmus hledá maximální párování v obecném grafu G. V souboru `edmonds.cpp` je k nalezení jeho funkční implementace (napsaná podle popisu ve Skriptíčkách z kombinatoriky [1]) pracující v čase $O(n^2(n+m))$, kde n je počet vrcholů a m počet hran grafu G. V následujícím textu jsou popsány zejména použité datové struktury.

Použití

Program `edmonds` přečte graf ze standardního vstupu v následujícím formátu a ve stejném formátu vypíše na standardní výstup graf s nalezeným maximálním párováním.

Na prvním řádku jsou tři celá nezáporná čísla oddělená mezerami: n, m a p, kde n je počet vrcholů, m počet hran a p je 0 nebo 1. Je-li $p = 1$, je graf zadaný včetně nějakého párování. Následuje m řádek, na každé z nich první dvě čísla $u, v \in \{0, \ldots, n-1\}$ značí vrcholy, mezi kterými vede daná hrana. Pokud je $p = 1$, je na řádce ještě třetí číslo: 1, pokud je hrana párovací, a 0 jinak.

Na vizualizaci řešení je možno použít program `graf-dot`. Ten výstup převede do jazyka dot, který slouží jako vstup pro nástroj `graphviz`[1], jímž byly generovány například obrázky v tomto textu.

Algoritmus

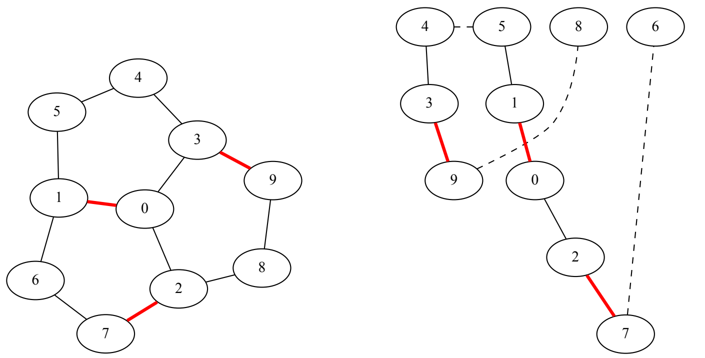
Edmondsův zahradní algoritmus najde maximální párování v obecném grafu. Začne s libovolným párováním a v každém kroku jej zvětší alespoň o jedna tak, že najde *volnou střídavou cestu* (cestu liché délky, jejíž koncové vrcholy nejsou spárované a pravidelně se na ní střídají párovací a nepárovací hrany) a párování na ní invertuje. Dá se dokázat (viz Skriptíčka [1]), že párování v grafu je maximální, právě když v něm neexistuje volná střídavá cesta.

Graf je reprezentován polem vrcholů, každý vrchol obsahuje pole odkazů na půlhrany. Půlhrana je pomocí pointerů spojena s odpovídající půlhranou u jiného vrcholu a se svým domovským vrcholem. Pro každou půlhranu si také pamatujeme, jestli je součástí párovací či nepárovací hrany.

* `matej.kocian@gmail.com`

[1] `http://graphviz.org`

V každém kroku algoritmu zkonstruujeme *Edmondsův les* tak, že ze všech volných vrcholů postupně *z jednoho po druhém* pustíme prohledávání do šířky. Z vrcholů na sudých hladinách ovšem půjdeme jen po nepárovacích hranách a naopak.

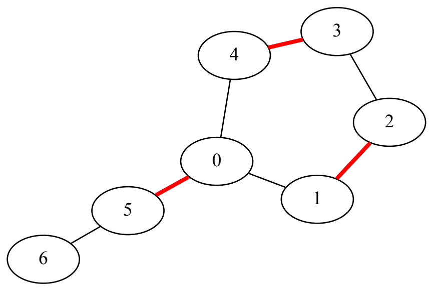


Obrázek 1: Graf s očíslovanými vrcholy a jemu odpovídající Edmondsův les. Červené hrany jsou párovací, čárkované vedou mezi sudými hladinami různých stromů a jejich nalezení by ve skutečnosti konstrukci lesa ukončilo.

Edmondsův les stihneme vybudovat v čase $O(n + m)$ (často ho ovšem ani nevybudujeme celý). Volné vrcholy nalezneme projitím všech a zkontrolováním, jestli jsou všechny vycházející hrany nepárovací. Pro každý vrchol si pamatujeme, zda leží na sudé či liché hladině a kterou hranou se jde k jeho rodiči. Díky tomu můžeme ve chvíli, kdy narazíme na hranu mezi sudými hladinami různých stromů Edmondsova lesa (a zjistíme tak, že graf obsahuje volnou střídavou cestu), vyšplhat v čase $O(n+m)$ do obou kořenů a po cestě invertovat párování.

Pokud narazíme na hranu mezi sudými hladinami jednoho stromu, našli jsme *květ* (lichou kružnici, na které se pravidelně střídají párovací a nepárovací hrany až na dvě nepárovací hrany vedle sebe; z vrcholu mezi nimi může vést *stonek* – střídavá cesta; viz obr. 2) a musíme jej zkontrahovat. To uděláme tak, že vytvoříme nový graf G_2, ve kterém bude tento květ zkontrahovaný.

Nejdříve po hranách do rodičů vystoupáme z obou vrcholů spojených onou hranou až do začátku stonku a u všech vrcholů, které cestou potkáme, si poznamenáme, že patří do květu. Každému vrcholu z G přiřadíme pozici `nove_cislo` odpovídajícího vrcholu v G_2; zkontrahovaný květ bude na pozici 0. Pak projdeme vrcholy G, vytvoříme odpovídající vrcholy v G_2 a přidáme hrany. Hranu vedoucí do květu přidáme jen tehdy, když výchozí vrchol není v květu a ještě z něj žádná hrana do květu nevede (to si pamatujeme v poli `uz_hrana`). Hranu vedoucí do vrcholu mimo květ přidáme jen tehdy, když je číslo výchozího vrcholu menší než číslo cílového. Hrany vedoucí z květu ignorujeme. Tímto způsobem žádnou hranu nepřidáme dvakrát. V každém případě vždy vytvoříme obě polo-



Obrázek 2: Příklad květu (červené hrany jsou párovací)

viny hrany naráz, abychom je mohli svázat pointery. Do jakého vrcholu uložit druhou půlhranu víme z pole `nove_cislo`. Pro každou novou hranu si také zapamatujeme odkaz na odpovídající hranu v G, abychom mohli změny v párování na G_2 snadno přenést do původního grafu. Celou konstrukci G_2 zvládneme v čase $O(n+m)$.

Následně rekurzivně zavoláme jeden krok Edmondsova algoritmu na G_2. Pokud se na něm podařilo párování vylepšit, upravíme G, jinak skončíme. Je-li květ v G_2 volný, jednoduše párování mimo květ okopírujeme pomocí odkazů na odpovídající hrany. Pokud do květu v G_2 vede párovací hrana e, musíme navíc párování na květu v G „pootočit“ tak, aby e tvořila začátek stonku a žádný vrchol tak nebyl spárovaný dvakrát (*přesadíme* květ). To celé stihneme v čase $O(n+m)$, dohromady tedy na každé úrovni rekurze trávíme tento čas a potřebujeme tolik paměti.

Hloubka rekurze nepřesáhne $n/2$, protože kontrahujeme kružnice a vždy tak ubydou alespoň dva vrcholy. Jeden krok algoritmu tedy trvá $O(n(n+m))$. Celkem se kroků provede nejvýše $n/2$, protože každý zvětší párování alespoň o jedna a párování nemůže obsahovat víc než $n/2$ hran. Maximální párování se podaří nalézt v čase $O(n^2(n+m))$.

Závěr

Naprogramovali jsme *nějakou* konkrétní implementaci Edmondsova algoritmu; je pravděpodobné, že algoritmus jde napsat snáze a rychlejší (s lepšími konstantami) při zachování časové složitosti $O(n^2(n+m))$. Navíc Tomáš Valla píše, že „pokud nebudeme Edmondsův algoritmus používat v inkrementální podobě a přepíšeme ho rovnou jako algoritmus na nalezení maximálního párování z prázdného, odpadnou tím některé opakované operace a lze ho použitím chytrých datových struktur urychlit na čas $O(n^3)$“ [1]. Nepochybně je zde tedy ještě prostor pro zlepšení.

Reference

[1] Valla, Tomáš a Matoušek, Jiří. *Kombinatorika a grafy I* [online]. 2005 [cit. 15. 3. 2013]. Dostupné z `http://kam.mff.cuni.cz/~valla/kg.pdf`.