7. CVIČENÍ Z ADS 1

Minimální kostry a vyvážené binární vyhledávací stromy

1 Rozcvička.

- Jak hledání minimální kostry ovlivní záporné hrany nebo záporné cykly?
- Co kdybychom chtěli najít kostru *maximální* namísto minimální?
- Jak se zbavit předpokladu o unikátních vahách hran?

2 Dynamická kostra.

Máme nalezenu minimální kostru a nyní chceme najít novou, pokud:

- z grafu odstraníme hranu,
- do grafu přidáme hranu,
- snížíme váhu hrany, nebo
- zvýšíme váhu hrany.

(Na změnu stačí lineární čas, umí se i amortizovaný čas $O(\text{poly}(\log n))$.)

3 Kostry s malými čísly.

Vymyslete algoritmus na nalezení minimální kostry grafu, v němž jsou váhy hran přirozená čísla z $\{1, \ldots, k\}$.

Řezové lemma. Nechť G je souvislý graf s unikátním ohodnocením hran a R elementární řez v G. Pak je nejlehčí hrana řezu R v minimální kostře.

Cyklové lemma. Nechť G je souvislý graf s unikátním ohodnocením hran a C cyklus v G. Pak nejtěžší hrana C neleží v minimální kostře.

Žluto-modrý algoritmus. Na začátku obarvíme všechny hrany černě. Dokud existuje alespoň jedna černá hrana, aplikujeme jedno z následujících pravidel:

- Zvolíme libovolně elementární řez R, jehož nejlehčí hrana e je černá, a obarvíme e modře.
- Zvolíme libovolně cyklus C, jehož nejtěžší hrana e je černá, a obarvíme e žlutě.

4 Všichni tři jsou žluto-modří.

Ukažte, že Jarníkův, Borůvkův i Kruskalův algoritmus se dají popsat jako speciální případy žluto-modrého algoritmu.

5 Správnost žluto-modrých algoritmů.

Dokažte cyklové lemma. Poté odvoďte, že libovolná varianta žluto-modrého algoritmu nalezne minimální kostru (pro graf s unikátními váhami).

6 Neunikátní lemmata.

Jak se změní řezové a cyklové lemma pro neunikátní váhy?

7 Další operace s BVS.

Rozmyslete si, jak implementovat v (libovolném) BVS následující operace (pro některé operace je nutné BVS trochu upravit). Jakou mají časovou složitost?

1. Operace následníka / předchůdce pro zadaný vrchol BVS (následník daného vrcholu s klíčem x je vrchol s nejmenším vyšším klíčem než x; podobně předchůdce).
2. Jak najít k-tý nejmenší prvek v BVS?
3. Nechť má každý vrchol kromě klíče přiřazenu i číselnou hodnotu (implementujeme tedy slovník). Jak najít součet hodnot přiřazených klíčům menších nebo rovných zadanému k?

(Bonus: Pokud si udržujete nějaké informace navíc, jak je udržovat při rotacích?)

8 Iterovaná následník.

Najdeme v BVS vrchol s minimálním klíčem (jak?) a poté $n - 1$ krát provedeme operaci nalezení následníka. Jaká bude celková časová složitost?

9 Dokonalé vyvážení.

Navrhněte algoritmus, který ze setříděného pole vyrobí v lineárním čase dokonale vyvážený BVS. (Pro každý vrchol se počet vrcholů v levém podstromu liší od počtu vrcholů v pravém podstromu maximálně o 1.)

Bonusové úlohy:

10 Kontrahující Borůvka.

Borůvkův algoritmus můžeme upravit, aby každý strom lesa udržoval zkontrahovaný do jednoho vrcholu. Iterace pak vypadá tak, že si každý vrchol vybere nejlehčí incidentní hranu, tyto hrany zkontrahujeme a zapamatujeme si, že patří do minimální kostry. Ukažte, jak tento algoritmus implementovat tak, aby běžel v čase $O(m \log n)$. Jak si poradit s násobnými hranami a smyčkami, které vznikají při kontrakci?

11 Rovinný Borůvka.

Jak rychle najde vhodně implementovaný algoritmus z předchozího příkladu minimální kostru v rovinném grafu? Naopak najděte (nerovinný) graf, na kterém algoritmus z předchozího příkladu poběží v čase $\Theta(m \log n)$.

12 Poměrová vyváženost.

Pro poměrově vyvážený BVS platí v každém vrcholu v, že $0.5 \leq |T_{\ell(v)}|/|T_{r(v)}| \leq 2$, kde $|T_{\ell(v)}|$ a $|T_{r(v)}|$ jsou velikosti podstromů levého a pravého syna. Ukažte, že poměrově vyvážený strom má logaritmickou hloubku.