8. CVIČENÍ Z ADS 1, ČTVRTEK 7.4. 10:40

Binární vyhledávací stromy (BVS)

1. *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?

a) Operace následníka / předchůdce pro zadaný vrchol BVS (následník daného vrcholu s klíčem x je vrchol s nejmenším vyšším klíčem než x; podobně předchůdce).
b) Jak najít k-tý nejmenší prvek v BVS?
c) 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?)

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

3. *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.)

4. *BVS-join.* Navrhněte algoritmus, který dostane dva BVS T_1 a T_2 a sloučí jejich obsah do jediného BVS v čase $O(|T_1| + |T_2|)$.

5. *BVS-split.* Pro zadané BVS T a hodnotu k navrhněte algoritmus, který T rozdělí na dva BVS T_1 a T_2, přičemž v T_1 jsou hodnoty menší nebo rovné k a v T_2 jsou hodnoty větší než k.

6. *Jen dva ukazatele.* Obvyklá reprezentace BVS v paměti potřebuje v každém vrcholu tři ukazatele (na levého a pravého syna a na otce). Jak si vystačit jen se dvěma ukazateli v každém vrcholu? Původní tři ukazatele by mělo jít spočítat pro libovolný vrchol v čase $O(1)$.

Bonusové úlohy:

7. *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.

8. *Poměrová vyváženost 2.* Uvažme následující strategii vyvažování po přidání nového klíče: Přidáme nový klíč do listu a pokud se pro nějaký vrchol v poruší podmínka poměrové vyváženosti, tak pro takové v nejbližší ke kořeni podstrom v přebudujeme na dokonale vyvážený. Ukažte, že amortizovaná složitost n operací přidání je $O(\log n)$.

9. *Dokonalé vyvážení „na místě“.* Navrhněte algoritmus běžící v lineárním čase, který zadaný BVS dokonale vyváží, ale použije pouze logaritmicky mnoho pomocné paměti. Zvládli byste to i s konstantně mnoho pomocné paměti?