

2. CVIČENÍ Z DATOVEK 1, ZS26/27

Amortizujeme! A připravujeme se na splay stromy

1. *Fronta pomocí dvou zásobníků.* Mějme dva zásobníky neomezené kapacity (vkládání/odebírání probíhá jen z vrchu zásobníku). Jak pomocí nich implementovat frontu (pouze s konstantní pomocnou pamětí)? Jakou amortizovanou složitost mají operace s frontou?

2. *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.)

3. *Tvrzení pro $BB[\alpha]$ stromy.* Dokažte následující: Jestliže podstrom vrcholu u po provedení operace Build má s_u vrcholů, pak další porušení vyvažovací podmínky pro vrchol u nastane nejdříve po $\Omega(s_u)$ přidání/smazání prvků v podstromu vrcholu u .

4. *Detail v potenciálu pro $BB[\alpha]$ stromy.* Připomeňte si potenciál pro amortizovanou analýzu $BB[\alpha]$ stromů. Co by se pokazilo, kdybychom v definici neudělali výjimku pro rozdíl 1?

5. *Příprava na splay stromy aneb rotujeme hledaný prvek do kořene.* Upravíme BVS (bez vyvažování) tak, že po vyhledání klíče x uzel s klíčem x vyrotujeme do kořene, tedy opakujeme následující: dokud má vrchol x rodiče p , provedeme jednoduchou rotaci mezi x a p . (Pro jednoduchost žádné klíče nevkládáme ani nemažeme.)

- Přístupy v rostoucím pořadí.* Začněte s libovolným binárním vyhledávacím stromem obsahujícím právě klíče $1, 2, \dots, n$, kde $n \geq 2$. Postupně proveďte přístupy ke klíčům $1, 2, \dots, n$, vždy v aktuálním stromu vzniklém po předchozím přístupu. Jak bude vypadat strom po celé posloupnosti? Závisí jeho výsledný tvar na počátečním stromu?
- Opakování rostoucí posloupnosti.* Začněte se stromem, který vznikl na konci předchozí úlohy, a znovu proveďte přístupy $1, 2, \dots, n$. Určete celkový počet rotací a časovou složitost pro tuto druhou rostoucí posloupnost.

6. *Oboustrané nafukovací/dynamické pole.* Zatím jsme přidávali (nebo odebírali) jenom na konci pole. Šlo by konstrukci upravit tak, abychom mohli přidávat i na začátek a zachovali jsme přitom složitost? A co kdybychom chtěli prvky z konce (či začátku) mazat?

7. *Okénková minima amortizovaně konstantně:* Na vstupu postupně přicházejí čísla v nekonečném proudu. Kdykoliv přijde další, vypište minimum z posledních k čísel. Cílem je dostat amortizovaně konstantní čas na jedno číslo.

8. *Víc než jen inkrement.* Uvažujme místo INC operaci $\text{ADD}(k)$, která k počítadlu přičte číslo k . Dokažte, že amortizovaná složitost této operace je $O(\log k)$.

9. *Dražší binární počítadlo.* Uvažme binární počítadlo, které umí přičítat jenom jedničku, ale cena závisí na pozici bitu. Ukažte, že když za přehození k -tého bitu platíme 2^k (a nultý bit je nejméně významný bit), pak posloupnost ℓ přičtení jedničky stojí $\mathcal{O}(\ell \cdot \log \ell)$.