

## 11. CVIČENÍ Z ADS 1, ČTVRTEK 28.4. 10:40

Hešování a amortizace

1. *Rozbití funkce.* Dostali jste hešovací funkci  $h : \mathcal{U} \rightarrow [m]$ . Pokud o této funkci nic dalšího nevíte, kolik nejvýše vyhodnocení funkce potřebujete, abyste našli  $k$ -tici prvků, které se všechny zobrazí do téže přihrádky?
2. *Dvojice s daným součtem.* Mějme množinu přirozených čísel a číslo  $x$ . Chceme zjistit, zda množina obsahuje dvojici prvků se součtem  $x$ .
3. *Binární počítadlo.* Mějme číslo  $n$  zapsané binárně (jako pole bitů). Jak dlouho trvá operace INC, která přičte k  $n$  jedničku? Jak dlouho trvá  $n$ -krát opakovaná operace INC, pokud začneme s počítadlem nastaveným na 0?
4. *Nafukování pole jinak.* Na přednášce jsme měli, že pokud opakovaně přidáváme prvky do hešovací tabulky (nebo jen pole), a vždy když dosáhneme jistého zaplnění, zdvojnásobíme velikost tabulky v čase lineárním s aktuálním počtem prvků, dosáhneme amortizované složitosti  $O(1)$  na vložení. Co kdybychom:
  - a) zvětšili velikost tabulky o konstantní počet prvků?
  - b) zvětšili velikost tabulky z  $m$  na  $m^2$ ?
5. *Fronta pomocí dvou zásobníků.* Mějte 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?
6. *Úprava lineárního přidávání.* Uvažujme hešování s otevřenou adresací řízené obecnější lineární posloupností  $h(x, i) = (f(x) + c \cdot i) \bmod m$ , kde  $c$  je konstanta nesoudělná s  $m$ . Srovnejte jeho chování s obyčejným lineárním přidáváním.
7. *Opravdové mazání.* Uvažujme hešování s otevřenou adresací a lineárním přidáváním. Mazání jsme na přednášce vyřešili pouze označením prvku za smazaný a pokud takto označíme  $\Omega(n)$  prvků, tabulku přebudujeme, což vede na amortizovanou složitost  $O(1)$ .

Navrhněte provedení mazání, které skutečně smaže prvek z tabulky v čase stejném jako přidání prvku a (neúspěšné) hledání. Proč to nebude fungovat pro dvojité hešování?

8. *Bloomův filtr*. Bloomův filtr je datová struktura pro přibližnou reprezentaci množiny. Skládá se z pole bitů  $B[1, \dots, m]$  a hešovací funkce  $h : \mathcal{U} \rightarrow [m]$ , přičemž operace jsou implementované takto:

- INSERT( $x$ ) nastaví  $B[h(x)] = 1$ ,
- MEMBER( $x$ ) otestuje, zda  $B[h(x)] = 1$ .

Vložme nyní do filtru nějakou  $n$ -prvkovou množinu  $M$ . Pokud  $x \in M$ , MEMBER( $x$ ) vždy odpoví správně. Pokud se ale zeptáme na  $x \notin M$ , může se stát, že  $h(x) = h(y)$  pro nějaké  $y \in M$ , a MEMBER( $x$ ) odpoví špatně. Spočítejte, s jakou pravděpodobností se to pro dané  $m$  a  $n$  stane (předpokládejte, že  $h$  je náhodná, nebo přesněji, že je náhodně vybraná z 1-univerzálního systému).

9. *Mimozemský narozeninový paradox*. Mějme planetu s  $m$  dny v roce a skupinu  $n$  obyvatel této planety, kteří mají rovnoměrně náhodně narozeniny v tamním roce. Zkuste co nejlépe spočítat pravděpodobnost, že dva mimozemšťané v této skupině mají narozeniny ve stejný den.