

11. cvičení

Datové struktury I, 13. 12. 2022

<https://iuuk.mff.cuni.cz/~chmel/2223/ds1/>

Úloha 1 (Delete v lineárním přidávání bez náhrobků)

Na přednášce jste viděli, jak implementovat operaci delete pro hashování s lineárním přidáváním pomocí náhrobků. Navrhněte, jak toto implementovat bez náhrobků.

Máte dvě možnosti jak to dělat: buď si u každého prvku nebudete pamatovat vůbec nic, nebo si u každého prvku budete pamatovat jak daleko je od svého hashe.

Úloha 2 (FKS [Fredman, Komlós, Szemerédi])

] Ukážeme si konstrukci (statické) hashovací tabulky pro podmnožinu S univerza \mathcal{U} , která nemá žádné kolize. Na přednášce jste měli konstrukci, která potřebovala $\Omega(n^2)$ paměti (přesněji paměťových buněk). My to zvládneme s lineárním počtem paměťových buněk (za předpokladu, že můžeme mít zcela náhodnou hashovací funkci, tu dokážeme sestrojít a samplovat v konstantním čase a že si ji dokážeme pamatovat v konstantním prostoru)¹.

Proces stavby tabulky bude probíhat následovně: budeme stavět dvě úrovně. V první úrovni si zcela náhodnou hashovací funkcí f rozdělíme prvky S do kyblíků B_1, \dots, B_n (a označíme si $b_i := |B_i|$). V druhé úrovni pak postavíme bezkolizní tabulku pomocí konstrukce z přednášky.

Začneme s první úrovní. V konstantním čase si vybereme náhodnou hashovací funkci $f : \mathcal{U} \rightarrow [n]$, a tou rozdělíme S do kyblíků. Toto opakujeme, dokud neplatí, že $\sum_{i=1}^n b_i^2 \leq \beta n$ pro $\beta = 4$.

Chceme ukázat, že tento krok budeme ve střední hodnotě opakovat nejvýše dvakrát. Označme jako C počet kolizí.

1. Určete $\mathbb{E}[C]$.
2. Určete C v závislosti na b_i .
3. Na základě předchozích dvou hodnot určete $\mathbb{E}[\sum_{i=1}^n b_i^2]$.
4. Aplikujte Markovovu nerovnost na náhodnou veličinu $X = \sum_{i=1}^n b_i^2$ s vhodnou hodnotou, abychom dostali požadovaný výsledek. (Taky se bude hodit střední hodnota geometrického rozdělení.)

Ve druhé úrovni pro každé $i \in [n]$ volíme v i -tém kyblíku univerzální hashovací funkci $g_i : \mathcal{U} \rightarrow [\alpha b_i^2]$ pro $\alpha = 2$. Toto opakujeme, dokud není prostá pro prvky v kyblíku B_i .

Označme jako C_x počet kolizí klíče $x \in B_i$ na druhé úrovni. Chceme ukázat, že pro kyblík i je střední hodnota počtu opakování volby g_i nejvýše 2.

1. Shora odhadněte $\mathbb{E}[C_x]$.
2. Použijte Markovovu nerovnost a union bound, abyste shora odhadli pravděpodobnost existence prvku aspoň s jednou kolizí.
3. Kolikrát budeme muset proces opakovat? ([Sem si vložte si svou oblíbenou poznámku o střední hodnotě geometrického rozdělení.])

Bonusové úlohy

Úloha 3 (Něco „praktičtějšího“)

Při hashování často počítáme modulo, ale to se přeloží do strojového kódu jako operace `idiv`, která je hodně drahá (latence pro 64 bitová čísla může být, dle procesoru, klidně i řádově desítky cyklů). Existují ale speciální prvočísla, tzv. Mersennova, pro které to jde rychleji. Ta mají tvar $p = 2^s - 1$.

Zkuste naimplementovat modulo Mersennovo prvočísla jen pomocí bitových operací (bitshift, bitwise and/or), sčítání, odčítání a porovnávání (obecně rychlých operací).

¹Totéž jde udělat s univerzální funkcí, tohle je jenom pro jednoduchost.