Notatki z AiSD. Nr 20 23 czerwca 2024

Słownik statyczny

IIUWr. II rok informatyki. *Opracował: Krzysztof Loryś*

1 Słownik statyczny

Ustalony zbiór n kluczy chcemy zapamiętać tak, by:

- struktura zajmowała n komórek pamięci,
- (oczekiwany) czas konstrukcji struktury był wielomianowy względem n,
- czas wykonywania instrukcji *find* był stały.

Nie chcemy wykonywać operacji *insert* i *delete*.

IDEA:

- stosujemy haszowanie dwupoziomowe,
- na pierwszym poziomie funkcja haszująca rozrzuca klucze do kubełków tak, by $\sum_{i=0}^{n-1} n_i^2 = O(n)$, gdzie n_i - liczba kluczy wrzuconych do kubełka i,
- na drugim poziomie haszujemy klucze, niezależnie w każdym kubełku, używając tablicy o rozmiarze n_i^2; haszowanie to jest bezkolizyjne,
- funkcje haszujące są brane losowo z uniwersalnej rodziny funkcji haszujących.

Lemat 1 (Nierówność Markowa) *Dla każdej zmiennej losowej X i dla każdego $t > 0$:*

$$Pr[|X| \geq t] \leq \frac{E[|X|]}{t}.$$

Fakt 1 *Z prawdopodobieństwem co najmniej 1/2 funkcja wybrana losowo z rodziny uniwersalnej bezkolizyjnie umieszcza $n = \sqrt{m}$ kluczy w tablicy m elementowej.*

UZASADNIENIE: W zbiorze n kluczy jest $< n^2/2$ par kluczy. Każda para koliduje z ppb $\leq 1/m$. Stąd oczekiwana liczba kolizji podczas wstawiania n kluczy jest mniejsza niż $n^2/(2m) < 1/2$. Stosujemy lemat Markowa dla $t = 1$. □

Lemat 2 *Jeśli do umieszczenia n kluczy w tablicy n elementowej użyjemy funkcji losowo wybranej z rodziny uniwersalnej, to z prawdopodobieństwem co najmniej $1/2$ zachodzi:*

$$\sum_{j=0}^{n-1} n_j^2 < 4n,$$

gdzie n_j oznacza liczbę kluczy umieszczonych w j-tym kubełku.

UZASADNIENIE:

Najpierw pokazujemy, że wartość oczekiwana sumy $\sum_{j=0}^{n-1} n_j^2$ jest mniejsza od $2n$, potem stosujemy nierówność Markowa dla $t = 4n$.

Chcemy obliczyć $E[\sum_{j=0}^{n-1} n_j^2]$. Umiemy policzyć:

- $E[\sum_{j=0}^{n-1} n_j]$. Ta suma jest równa n - liczbie wszystkich kluczy w słowniku.

- $E[\sum_{j=0}^{n-1} \binom{n_j}{2}]$. Ta suma jest równa liczbie wszystkich kolizji. Ponieważ każda para kluczy koliduje z ppb'stwem nie większym od $1/n$ (tu n jest także wielkością tablicy), więc oczekiwana liczba kolizji jest nie większa od $\frac{n(n-1)}{2} \cdot \frac{1}{n} = \frac{n-1}{2}$.

Ale
$$E\left[\sum_{j=0}^{n-1} \binom{n_j}{2}\right] = E\left[\sum_{j=0}^{n-1} \frac{n_j^2 - n_j}{2}\right],$$
więc
$$E\left[\sum_{j=0}^{n-1} n_j^2\right] = 2E\left[\sum_{j=0}^{n-1} \binom{n_j}{2}\right] + E\left[\sum_{j=0}^{n-1} n_j\right] \leq 2\frac{n-1}{2} + n < 2n.$$
□

Reasumując:

- **Pamięć**. Potrzebujemy (dla dobrze wylosowanych funkcji):
  - nie więcej niż $4n$ komórek na tablice wtórne,
  - trzy komórki na parametry funkcji pierwotnej (jedną na p, jedną na a, jedną na b),
  - dodatkowo na każdą tablicę wtórną 3 komórki: jedną na rozmiar tablicy; dwie na parametry funkcji; czyli w sumie $3n$ komórek.
- **Czas tworzenia struktury.**
  - Oczekiwana liczba losowań funkcji pierwotnej jest nie większa od 2. Sprawdzenie, czy wylosowana funkcja jest dobra, wymaga obliczenia jej wartości dla wszystkich n kluczy. To daje się zrobić w czasie $O(n)$.
  - Oczekiwana liczba losowań wtórnej funkcji losowej (dla j-tego kubełka) jest nie większa od 2. Czas sprawdzenia, czy jest dobra, jest $O(n_j)$. Stąd oczekiwany czas związany z losowaniem wszystkich funkcji wtórnych jest $O(n)$.
- **Czas instrukcji *find*.** Jest stały - wystarczy bowiem wyliczyć wartości dwóch funkcji haszujących.