BLOOMOVY FILTRY

[první struktura tohoto typu: Bloom 1970]

Úloha: Chceme reprezentovat množinu $X \subseteq \mathcal{U}$, $|X|=n$ a odpovídat na dotazy "$a \in X$?"
... ovšem máme málo paměti ⇒ spokojíme se s přibližným řešením.
Možné chyby: false positive (tvrdíme, že $a \in X$, pro $a \notin X$)
false negative (opačně) [ty nakonec žádné nebudou]
} chceme omezit pravděpodobnost

První pokus: Pořídíme si pole $B[0 \ldots m-1]$ ~~m~~ bitů a hešovací fci $h: \mathcal{U} \to [m]$ z 1-univ. systému.
Začneme se samými 0, postupně nastavíme bity $h(x_1), \ldots, h(x_n)$.

- Find(a) vrátí TRUE, pokud $B[h(a)] = 1$
  - pro $a \in X$ vždy odpoví správně
  - pro $a \notin X$ false positive s pstí $\leq \frac{n}{m}$ (z 1-universality)

  pro $m = c \cdot n$ je psť. chyby $\leq \frac{1}{c}$
- Insert(a) nastaví $B[h(a)] \leftarrow 1$
- Delete neumíme provést

Vylepšení: Pořídíme si k filtrů s $m = 2n$ a různými heš. funkcemi (nezávislými).
Find odpoví TRUE, pokud najdeme ve všech filtrech:
- pro $a \in X$ odpovíme správně
- pro $a \notin X$ je $\Pr[\text{false positive}] \leq 2^{-k}$

Tedy pro ~~chybu~~ Pr chyby ε potřebujeme prostor $2n \cdot \log \frac{1}{\varepsilon}$.

Podrobnější analýza (předpokládá dokonale náhodné heš. fce)
- fixujeme celkový prostor $M = mk$, hledáme optimální k
- $\underbrace{\Pr[B_i[j]=0]}_{\text{označme } p'} = \left(1-\frac{1}{m}\right)^n = \left(1-\frac{k}{M}\right)^n \approx \underbrace{e^{-\frac{kn}{M}}}_{p}$
- $\Pr[\text{false positive}] = (1-p')^k \approx (1-p)^k = \left(1-e^{-\frac{kn}{M}}\right)^k = e^{k \cdot \ln(1-p)}$
- $k \cdot \ln(1-p) = \left(-\frac{M}{n}\right) \underbrace{\ln p \cdot \ln(1-p)}$
  └ má na intervalu $(0,1)$ min. pro $p = \frac{1}{2}$
  - chceme tedy $e^{-\frac{kn}{M}} = \frac{1}{2}$ ... $-\frac{kn}{M} = -\ln 2$ ... $k = \underbrace{\ln 2}_{\doteq 0.69} \cdot \frac{M}{n}$
  - pak $\Pr[\text{f.p.}] = 2^{-k}$

→ opačně: pro $\Pr[\text{f.p.}] \leq \varepsilon$ chceme $k := \left\lceil \log \frac{1}{\varepsilon} \right\rceil$,
tedy $M = \frac{kn}{\ln 2} \doteq \log \frac{1}{\varepsilon} \cdot n \cdot \frac{1}{\ln 2} \doteq 1.44 \cdot n \cdot \log \frac{1}{\varepsilon}$

$\log e \doteq 1.44$

Dolní odhad (bez důkazu): Je třeba alespoň $n \cdot \log \frac{1}{\varepsilon}$ bitů
⇒ náš B. filtr je ~~jen~~ max. 1.44-krát horší než optimum.

Další operace: • Máme-li pro $X, Y \subseteq \mathcal{U}$ filtry B_X a B_Y se stejným m, můžeme snadno spočíst $B_{X \cup Y} = B_X \vee B_Y$, $B_{X \cap Y} = B_X \wedge B_Y$

- z počtu 1 ve filtru lze odhadnout velikost množiny:
  - víme, že $\Pr[\text{bit je } 0] \approx e^{-\frac{kn}{M}}$
  - proto $\mathbb{E}[\#0] \approx M \cdot e^{-kn/M}$ … navíc $\#0$ je silně koncentrovaný kolem své $\mathbb{E}$ (bez důkazu)
- též lze vše hešovat do 1 společné tabulky (tedy $\forall i\; h_i : \mathcal{U} \to [M]$) … asymptoticky stejné

Počítací Bloomovy filtry [Fan et al. 2000]

- chceme umět i Delete
- v poli jsou místo 1-bit. indikátorů b-bitová počítadla
- Insert zvyšuje o 1, Delete snižuje o 1, Find testuje nenulovost
- POZOR: jakmile počítadlo dopočítá do max. hodnoty $(2^b - 1)$, musí tak navěky zůstat!

$$\hookrightarrow \Pr[C_i[j] = t] = \binom{n}{t} \cdot \left(\frac{1}{m}\right)^t \left(1 - \frac{1}{m}\right)^{n-t}$$

j-té počítadlo v i-tém filtru

předpokládáme plně náhodné funkce h_i

$$\Pr[C_i[j] \geq t] \leq \binom{n}{t}\left(\frac{1}{m}\right)^t \leq \left(\frac{ne}{t}\right)^t \cdot \left(\frac{1}{m}\right)^t = \left(\frac{ne}{mt}\right)^t \leq \left(\frac{e \cdot \ln 2}{t}\right)^t$$

aspoň 1 t-tice se zahešuje do j-té přihrádky, zbývající prvky mohou kamkoliv

std. odhad $\binom{n}{t}$, viz kapitoly z DM

už víme, že chceme $m \geq n \cdot \ln 2$

$$\Pr[\exists j : C_i[j] \geq t] \leq m \cdot \Pr[C_i[j] \geq t] \leq m \cdot \left(\frac{e \cdot \ln 2}{t}\right)^t$$

pro $t = 15$ je to $\doteq 3 \cdot 10^{-14}$

$\Rightarrow$ pro takřka jakékoli rozumné m stačí 4-bitová počítadla

„Bloomier filtry" – reprezentace funkcí [původně Fan, Cao, Broder 2000] [zde vylepšeno dle Charles, Chellapilla 2008]

- Máme funkci $f : X \to [2^r]$ definovanou na $X \subseteq \mathcal{U}$
- Chceme sestrojit funkci $g : \mathcal{U} \to [2^r]$ takovou, že $g \restriction X = f$ (mimo X může dávat cokoli)
- Inspirace Kukaččím hešováním:
  - pole $R[0 \ldots m-1]$ r-bitových hodnot
  - 2 hešovací funkce $h_1, h_2 : \mathcal{U} \to [m]$
  - Find vrátí $R[h_1(x)] \oplus R[h_2(x)]$ ($\oplus$ = XOR)
- Pro danou X najdeme h_1, h_2 t.ž. kukaččí graf je acyklický (pro $m \geq 2(1+\varepsilon)n$ to nastane s pstí aspoň konst. $\Rightarrow$ stačí průměrně $O(1)$ pokusů)
- Pak každou komponentu prohledáme do hloubky. Kořeni nastavíme $R[-] \leftarrow 0$, ostatním vrcholům dopočítáme, aby vyšly XORy.
- Stačí nám $2(1+\varepsilon)rn$ bitů paměti (konstanty lze zlepšit použitím více heš. fcí).