9. cvičení

Datové struktury I, 4. 12. 2025 https://iuuk.mff.cuni.cz/~chmel/2526/ds1/

Úloha 1 (Rolling hash je d-univerzální)
Pro prvočíslo p a délku vektoru d definujeme třídu hashovacích funkcí $\mathcal{R} = \{h_a \colon a \in \mathbb{Z}_p\}$, kde $h_a(x) = \sum_{i=0}^{d-1} x_{i+1}a^i$, a všechno počítáme modulo p. (Bereme $x \in \mathbb{Z}_p^d$, indexujeme od jedničky.)
Dokažte, že tato třída hashovacích funkcí je $(d-1)$-univerzální.
Hint: *Základní věta algebry říká, že každý polynom stupně d má nejvýše d kořenů – zkuste tam nějaký najít.*

Řešení
Zajímá nás pravděpodobnost, že pro $x \neq y \in \mathbb{Z}_p^d$ máme $h_a(x) = h_a(y)$. To je ekvivalentní tomu, že $h_a(x) - h_a(y) = 0$, a to můžeme přepsat z definice na $\sum_{i=0}^{d-1}(x_{i+1} - y_{i+1})a^i$, což je polynom v proměnné a se stupněm nejvýše $d-1$. Ze základní věty algebry pak máme, že tento polynom má nejvýše $d-1$ kořenů, a právě tyto kořeny určují funkce h_a, ve kterých se x a y zahashují na téže místo. Celkem tedy máme pravděpodobnost kolize x a y $\frac{d-1}{p}$, a tedy máme $(d-1)$-univerzalitu.

Úloha 2 (Vyhledávání jehly v textu)
Vymyslete algoritmus na nalezení všech výskytů podřetězce x délky n v textu T délky m pomocí hashování, který běží v průměrném čase (tj. ve střední hodnotě) $\mathcal{O}(n + m + k \cdot n)$, kde k je počet výskytů x v T.

Řešení
Rabin-Karp s Rolling hashem, čas: máme m času na projití řetězce, v každém kroku uděláme konstantní úpravu hashe a zkontrolujeme, že není stejný. Pokud je hash stejný, zkontrolujeme celý string. Protože máme hashování do nějakého $\mathbb{Z}_p$, pravděpodobnost kolize je d/p, a tedy celkem máme ve střední hodnotě asi $(k + md/p)$ kolizí. Pokud ale zvolíme $p > m \cdot d$ (nebo obecně stačí $p \in \Omega(m \cdot d)$), máme konstantně mnoho falešných kolizí ve střední hodnotě.

Úloha 3 (Přetečení v počítacím Bloomově filtru)
Uvažme počítací Bloomův filtr s maximální hodnotou jednoho počítadla ℓ. Tento Bloomův filtr bude mít m políček s počítadly, a budeme používat k zcela náhodných hashovacích funkcí. Určete pravděpodobnost toho, že nám tento Bloomův filtr přeteče (tedy budeme mít aspoň jedno počítadlo, které se dostalo na hodnotu ℓ). Pro jednoduchost můžete začít nejprve s $k = 1$, a potom zobecnit pro libovolné k.

Řešení
Pro přetečení je potřeba, aby se nám na nějaký index trefilo alespoň ℓ přičtení jedničky. Začneme s pravděpodobností, že se tam trefilo *právě* ℓ přičtení: $\binom{nk}{\ell} \cdot (\frac{1}{m})^\ell \cdot (1 - \frac{1}{m})^{nk-\ell}$. Tedy pravděpodobnost, že máme alespoň ℓ přičtení je přesně $1 - \sum_{i=1}^{\ell-1} \binom{nk}{i} \cdot (\frac{1}{m})^i \cdot (1 - \frac{1}{m})^{nk-i}$, ale můžeme ji shora odhadnout jako $\binom{nk}{\ell} \cdot (\frac{1}{m})^\ell$, protože aspoň ℓ věcí se musí zahashovat, a . Díky odhadu $\binom{nk}{\ell} \leq (kne/\ell)^\ell$ můžeme odhadnout celou pravděpodobnost jako $(\frac{nke}{\ell m})^\ell$, a z přednášky víme, že optimální volba m je cca $kn/\ln 2$, a tedy máme pravděpodobnost cca $(e \ln 2/\ell)^\ell$, a na odhad přes všechna počítadla použijeme union bound.

Úloha 4 (Bitová krize)
Máte Bloomův filtr s bitovým polem délky $m = 2^b$. Bohužel ale přišla krize, RAMky skokově zdražily[1] a vaše RAMka zrovna vyhořela (ale naštěstí předtím zvládla dumpnout všechno, co v ní bylo, na disk). Koupíte si tedy novou, ale protože je drahá, můžete si dovolit jenom 2^{b-1} bitů. Zároveň ale nechcete přijít o váš filtr – umíte ho nějak upravit, aby fungoval i s poloviční pamětí?

Řešení
Vezmeme OR první a druhé poloviny, a při dalších operacích všechny hashe vymodulíme 2^{b-1}.

Úloha 5 (Operace na Bloomových filtrech)
Uvažme dva jednopásové Bloomovy filtry s poli F_1, F_2, které mají stejnou délku, používají stejné hashovací funkce (potenciálně i více) a reprezentují množiny A_1, A_2. Uvažme Bloomův filtr s polem F_3, které vznikne tak, že $F_3[i] := F_1[i] \wedge F_2[i]$.

a) Je F_3 funkční Bloomův filtr pro množinu $A_1 \cap A_2$? (Tedy chceme, aby F_3 nikdy chybně neodpověděl „ne“.)

[1] Podobnost se skutečnými událostmi je čisté náhodná.

b) Je F_3 identický s Bloomovým filtrem, který by vzniknul postupným přidáváním prvků z množiny $A_1 \cap A_2$?

c) Umíte tento postup upravit i pro počítací filtry?

d) A co pro sjednocení? (A počítací filtry?)

Řešení a) Ano, pokud je $x \in A_1 \cap A_2$, tak jeho konkrétní místa budou vždycky jedničky.

b) Ne, stačí mít $c \neq d, A_1 = \{c\}, A_2 = \{d\}$, že $\exists i, j \colon h_i(c) = h_j(d)$, pak tento index bude v obou polích 1, ale $A_1 \cap A_2 = \emptyset$, a tedy i Bloomův filtr bude prázdný.

c) Místo ANDu použijeme minimum.

d) Použijeme OR, respektive + se zařiznutím na maximum.

Bonusové úlohy

Úloha 6 (Dosáhneme?)
Sestrojte pravděpodobnostní algoritmus[2] pro rozhodnutí dosažitelnosti v neorientovaném grafu (tedy odpověď na otázku „existuje cesta z u do v?“), který selže s pravděpodobností nejvýše 1/4, a potřebuje jenom logaritmický prostor.
Technická poznámka k logaritmickému prostoru: když máme algoritmus běžící v sublineárním prostoru, počítáme jej tak, že máme vstup na „read-only“ disku, kde k němu můžeme přistupovat, ale nemůžeme jej upravovat (ale podle potřeby si jej můžeme bez problému kopírovat).[3]
Pravděpodobně se vám bude hodit následující tvrzení: když v n-vrcholovém souvislém neorientovaném grafu vyjdeme z vrcholu u, a budeme v každém kroku uniformně náhodně vybírat ze všech sousedů vrchol, kam půjdeme v dalším kroku, střední hodnota doby než navštívíme jiný vrchol v, je nejvýše $2n^3$.
Taky se může hodit připomenout si Markovovu nerovnost: Buď X nezáporná náhodná veličina. Pak $\forall \varepsilon > 0$ platí $P[X \geq \varepsilon] \leq \frac{\mathbb{E}[X]}{\varepsilon}$. Ekvivalentně pro jakékoliv $d \geq 1$, $P[X \geq d \cdot \mathbb{E}[X]] \leq \frac{1}{d}$.

Řešení
Budeme náhodně chodit $8n^3$ kroků, pokud jsme našli, tak řekneme „dosažitelný“, pokud ne, tak řekneme „nedosažitelný“. Z Markovovy nerovnosti plyne, že pravděpodobnost neúspěchu u dosažitelného vrcholu je maximálně 1/4, a nedosažitelnost vždycky řekneme správně.

Fun fact. Octomliky mají vyvinutou jakousi verzi Bloomova filtru se zapomínáním pro pachy. Více viz `https://www.pnas.org/doi/10.1073/pnas.1814448115` – Dasgupta, Sheehan, Stevens, Navlakha: A neural data structure for novelty detection, *Proc. Natl. Acad. Sci. U.S.A.* 115 (51) 13093-13098.

[2] Pravděpodobnostní algoritmy jste asi už potkali, ale zkráceně: bude se jednat o algoritmus, který se bude moct v každém kroku rozhodnout náhodně z několika možností

[3] Technicky tuto třídu definujeme na Turingových strojích, kde máme read-only vstupní pásku, a druhou pracovní pásku. Do prostorové složitosti se nám počítá jenom pracovní páska.