7. cvičení

Datové struktury I, 15. 11. 2024 https://iuuk.mff.cuni.cz/~chmel/2425/ds1/

Cachování

Úloha 1 (LRU vs OPT)
Jak velký může být poměr počtu výpadků LRU a OPT strategií, používají-li stejně velkou cache? (Na začátku přepokládejte prázdnou cache.)

Úloha 2 (Jinak velká transpozice)
V cache-oblivious algoritmu[1] jsme tak nějak předpokládali, že velikost matice je mocnina dvojky (tj. je velikosti $2^n \times 2^n$). Co se stane, pokud tomu tak není? Dokažte, že pro matici $n \times n$ jsou všechny podmatice v průběhu *skoro čtvercové*, tedy počet řádků a sloupců se liší nejvýše o jedna. Potom nahlédněte, že tato vlastnost algoritmus nijak nerozbije.

Úloha 3 (Doslovný transpose-and-swap)
V cache-oblivious algoritmu je snadnou chybou provedení transpose-and-swap doslova. Tedy, nejdřív obě podmatice rekurzivně ztransponujeme, a pak je prohodíme. Zanalyzujte *časovou* složitost tohoto algoritmu a poté i I/O složitost (tedy počet přenesených bloků).

Příprava na hashování: opakování pravděpodobnosti

Úloha 4 (Lehké opakování pravděpodobnosti)
Matt střílí, a snaží se trefit se do terče. Protože je začátečník, pravděpodobnost, že se trefí, je $p \in (0, 1]$, a je nezávislá na jakýchkoliv jeho předchozích pokusech. Nechť X je náhodná veličina, která označuje počet pokusů do Mattovy první trefy (včetně). Ukažte, že $\mathbb{E}[X] = \frac{1}{p}$.

Úloha 5 (Pravděpodobnost kolize)
Ukažte, že v hashovací tabulce velikosti $m = n^2$ s n prvky dojde ke kolizi s pravděpodobností maximálně $\frac{1}{2}$, předpokládáme-li zcela náhodnou hashovací funkci. (Zkuste to dokázat bez použití přímo padnoucího pozorování z přednášky.)

Úloha 6 (Pevné body permutací)
Mějme uniformně náhodnou permutaci na n prvcích. Určete střední hodnotu počtu pevných bodů této permutace.

Úloha 7 (Černá krabička)
Dostali jste hashovací funkci $h : \mathcal{U} \to [m]$. Pokud o této funkci nevíte nic dalšího, kolik vyhodnocení funkce potřebujete, abyste zaručeně našli k-tici prvků, které se všechny zobrazí do téže přihrádky?

Užitečné pojmy

Tvrzení (Union bound)**.** Pro jevy A_1, A_2 platí, že $P[A_1 \cup A_2] \leq P[A_1] + P[A_2]$.

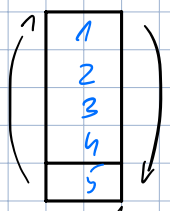
Tvrzení (Linearita střední hodnoty)**.** Pro náhodné veličiny X, Y a $\alpha, \beta \in \mathbb{R}$ platí: $\mathbb{E}[\alpha X + \beta Y] = \alpha\mathbb{E}[X] + \beta\mathbb{E}[Y]$.

Definice (Indikátor, nezávislost náhodných veličin)**.** Nechť A je jev v diskrétním pravděpodobnostním prostoru. Potom indikátor A je náhodná veličina I_A definovaná: $I_A(\omega) = 0 \Leftrightarrow \omega \notin A$, jinak $I_A(\omega) = 1$.
Náhodné veličiny X, Y na diskrétním pravděpodobnostním prostoru $(\Omega, 2^\Omega, P)$ jsou nezávislé, pokud $\forall \alpha, \beta \in \mathbb{R}$ jsou jevy $\{\omega \in \Omega : X(\omega) \leq \alpha\}, \{\omega \in \Omega : Y(\omega) \leq \beta\}$ nezávislé.

[1] Pokud si ho nepamatujete, připomeňte si ho, a pokud chcete, zkuste si odsimulovat, co se tam vlastně děje, na 8×8 matici s velikostí bloku $B = 3$.

Úloha 1 (LRU vs OPT)
Jak velký může být poměr počtu výpadků LRU a OPT strategií, používají-li stejně velkou cache? (Na začátku přepokládejte prázdnou cache.)

Cache velikosti P, blok velikosti B, celkem V bloků.
Nejhorší je, když LRU vyhodí číslo těsně před přečtení (to OPT nedělá)



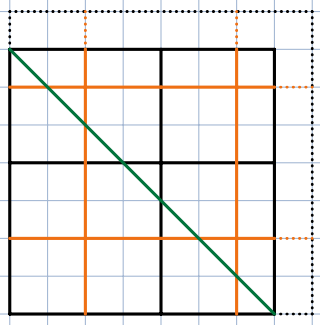
sekvenční čtení

LRU -> 1 výpadek / request
OPT -> 1 výpadek / V requestů
(V = velikost cache)

Poměr je $1/V$

Úloha 2 (Jinak velká transpozice)
V cache-oblivious algoritmu[1] jsme tak nějak předpokládali, že velikost matice je mocnina dvojky (tj. je velikosti $2^n \times 2^n$). Co se stane, pokud tomu tak není? Dokažte, že pro matici $n \times n$ jsou všechny podmatice v průběhu *skoro čtvercové*, tedy počet řádků a sloupců se liší nejvýše o jedna. Potom nahlédněte, že tato vlastnost algoritmus nijak nerozbije.

-> pokud se do cache vejde 2^4, vejde se určitě i tahle oříznutá

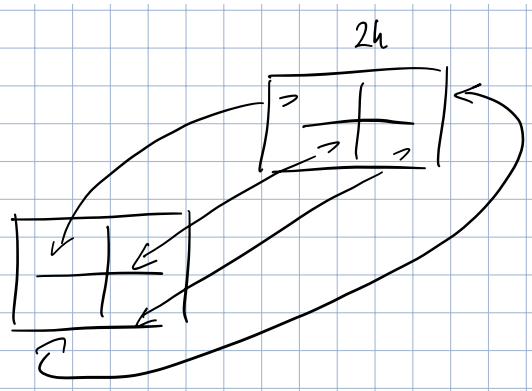


$$\left\lfloor \frac{n}{2} \right\rfloor \leq \frac{n}{2} \leq \left\lceil \frac{n}{2} \right\rceil$$

$$\left\lceil \frac{n}{2} \right\rceil - \left\lfloor \frac{n}{2} \right\rfloor \leq 1$$

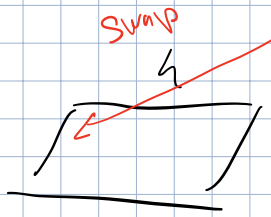
Úloha 3 (Doslovný transpose-and-swap)
V cache-oblivious algoritmu je snadnou chybou provedení transpose-and-swap doslova. Tedy, nejdřív obě podmatice rekurzivně ztransponujeme, a pak je prohodíme. Zanalyzujte *časovou* složitost tohoto algoritmu a poté i I/O složitost (tedy počet přenesených bloků).

$N = n^2 \quad T(N) = 4T(N/4) + \frac{N}{2} \doteq O(N \log n)$
$= O(n^2 \log n)$



transpose $\sim O(n)$
musím projít všechny prvky

T a S seq $\sim O(n^2)$



$\sim O(n)$
musím projít všechny prvky

Transpose nyní swap paralelně
$\sim O(n)$
stačí mi dva chlívky navíc a zbytek může dobrat na místě.

Úloha 4 (Lehké opakování pravděpodobnosti)
Matt střílí, a snaží se trefit se do terče. Protože je začátečník, pravděpodobnost, že se trefí, je $p \in (0,1]$, a je nezávislá na jakýchkoliv jeho předchozích pokusech. Nechť X je náhodná veličina, která označuje počet pokusů do Mattovy první trefy (včetně). Ukažte, že $\mathbb{E}[X] = \frac{1}{p}$.

$$\mathbb{E}(X) = \sum_{k=1}^{\infty} k \cdot p \cdot (1-p)^{k-1} = p \cdot \underbrace{\sum_{k=1}^{\infty} k \cdot (1-p)^{k-1}}_{\text{wolframalpha spočítal}} = p \cdot \frac{1}{p^2} = \frac{1}{p}$$

Úloha 5 (Pravděpodobnost kolize)
Ukažte, že v hashovací tabulce velikosti $m = n^2$ s n prvky dojde ke kolizi s pravděpodobností maximálně $\frac{1}{2}$, předpokládáme-li zcela náhodnou hashovací funkci. (Zkuste to dokázat bez použití přímo padnoucího pozorování z přednášky.)

Šance jednoho políčka: $1/n^2$

Šance obsazenosti políčka: $n/n^2 = \frac{1}{n}$

$$P(\text{kolize}) = P\left[\bigcup_{i \neq j} h(i) = h(j)\right] \overset{\text{union bound}}{\leq} \sum_{\substack{i \neq j \\ [n]}}^{n} \frac{1}{n^2} = \binom{n}{2} \cdot \frac{1}{n^2} \leq \frac{n^2}{2} \cdot \frac{1}{n^2} = \frac{1}{2}$$

Úloha 7 (Černá krabička)
Dostali jste hashovací funkci $h : \mathcal{U} \to [m]$. Pokud o této funkci nevíte nic dalšího, kolik vyhodnocení funkce potřebujete, abyste zaručeně našli k-tici prvků, které se všechny zobrazí do téže přihrádky?

$m \cdot (k-1) + 1$ → nejlepší případ, kdy jsem dostal jen „k-1" kolizí. Pak stačí lib. další prvek a budou mít k kolizí.