Cvičení 4

Příklad 1: Připomeňte si, co je to ***hashovací tabulka*** a jak jsou implementovány jednotlivé základní operace na této datové struktuře.

Uvažujme variantu hashovací tabulky, ve které jsou prvky umísťovány do samostatných zřetězených seznamů, jejichž prvky jsou alokovány mimo pole se sloty, které obsahuje pouze ukazatele na začátky jednotlivých seznamů. Předpokládejme, že toto pole se sloty má velikost m, tj. jednotlivé sloty jsou indexovány čísly $0, 1, \ldots, m-1$.

Řekněme, že tato tabulka byla na začátku prázdná, a poté do ní bylo postupně přidáno n prvků

$$x_1, x_2, \ldots, x_n\,.$$

(Předpokládejme, že nově přidávaný prvek byl vždy přidáván na začátek seznamu v příslušném slotu.)

Pro jednoduchost budeme předpokládat, že daná hashovací tabulka používá nějakou hashovací funkci h, jejíž hodnotu je možné spočítat v čase $\mathcal{O}(1)$, a že se tato funkce se chová „ideálně“ v tom smyslu, že pravděpodobnost umístění prvku do daného slotu je stejná pro všechny sloty, tj. že pro každý z prvků $x_1, x_2, \ldots, x_n$ je pravděpodobnost toho, že prvek x_i bude umístěn do slotu j (kde $0 \leq j < m$), rovna $1/m$.

Hodnota $\alpha = n/m$ udává průměrnou délku jednoho seznamu, a udává tak „míru“ zaplněnosti dané hashovací tabulky.

Analyzujte časovou složitost vyhledání prvku v hashovací tabulce v nejhorším a v průměrném případě pro následující dva případy (odhady složitostí vyjádřete v závislosti na n, m či α):

a) hledaný prvek se v tabulce nenachází, tj. hledaný prvek y je různý od všech prvků $x_1, x_2, \ldots, x_n$,

b) hledaný prvek se v tabulce nachází, tj. hledaný prvek je jedním z prvků $x_1, x_2, \ldots, x_n$ (předpokládejte, že pro každé $i \in \{1, 2, \ldots, n\}$ je stejně pravděpodobné, že se jedná o prvek x_i).

Poznámka: Při řešení můžete vycházet z analýzy popsané v sekci 11.2 (*Hash Tables*) v podsekci *Analysis of hashing with chaining* v kapitole 11 (*Hash Tables*) v knize T. Cormen, C. Leiserson, R. Rivest, C. Stein: *Introduction to Algorithms* (3rd edition), The MIT Press, 2009.

Příklad 2: Připomeňte si, co je to ***binární vyhledávací strom*** a jak jsou implementovány jednotlivé základní operace na této datové struktuře. Uvažujte binární vyhledávací stromy v základní nejjednodušší podobě, kde není použito žádné vyvažování.

Předpokládejme, že daný strom T vznikl z prázdného stromu provedením sekvence n operací INSERT s posloupností prvků

$$x_1, x_2, \ldots, x_n,$$

které byly postupně do stromu přidány. Předpokládejme dále pro jednoduchost, že prvky $x_1, x_2, \ldots, x_n$ jsou navzájem různé, a že každá jejich permutace je stejně pravděpodobná.

Výška stromu je délka jeho nejdelší větve (vyjádřená počtem hran).

Cílem je analyzovat, jaká bude výška vytvořeného stromu v průměrném případě, a ukázat, že bude v $\Theta(\log n)$.

Pokud tedy zavedeme náhodnou proměnnou X_n udávající výšku stromu o n vrcholech, cílem je ukázat, že pro střední hodnotu $E[X_n]$ platí $E[X_n] \in \Theta(\log n)$.

Detailně si promyslete jednotlivé kroky důkazu, který je uveden v sekci 12.4 (*Randomly built binary search trees*) v kapitole 12 (*Binary Search Trees*) knihy T. Cormen, C. Leiserson, R. Rivest, C. Stein: *Introduction to Algorithms* (3rd edition), The MIT Press, 2009.

Tento důkaz postupuje v několika krocích:

- Zavedou se další pomocné náhodné proměnné Y_n, kde $Y_n = 2^{X_n}$. (Intuitivně hodnota Y_n odpovídá počtu listů dokonale vyváženého stromu, který bychom dostali, pokud bychom v daném stromě o n vrcholech doplnili a prodloužili všechny větve tak, aby byly všechny stejně dlouhé, jako nejdelší větev původního stromu.)

  Je možné rekurzivně vyjádřit omezení na střední hodnotu $E[Y_n]$ shora: $E[Y_0] = 0$, $E[Y_1] = 1$ a pro $n > 1$ je možné spočítat omezení shora na hodnotu $E[Y_n]$ z hodnot omezujících shora hodnoty $E[Y_0], E[Y_1], \ldots, E[Y_{n-1}]$.

- Z daného rekurzivního vyjádření funkce, která shora omezuje hodnoty $E[Y_n]$, je pak možné odvodit (o něco hrubší) nerekurzivně vyjádřený odhad, který shora omezuje hodnotu $E[Y_n]$.

- Z výše uvedeného omezení hodnoty $E[Y_n]$ je pak už relativně snadné odvodit, že $E[X_n] \in \Theta(\log n)$.

Příklad 3: Posloupnost

$$\frac{1}{1}; \frac{1}{2}; \frac{1}{3}; \frac{1}{4}; \frac{1}{5}; \frac{1}{6}; \frac{1}{7}; \frac{1}{8}; \frac{1}{9}; \ldots$$

se označuje jako ***harmonická řada***.

Součet prvních n členů této řady se označuje jako n-té ***harmonické číslo***:

$$H_n = \sum_{i=1}^{n} \frac{1}{i}$$

Dokažte, že pro každé harmonické číslo H_n platí

$$\ln n + \frac{1}{n} < H_n < \ln n + 1$$

kde ln označuje přirozený logaritmus, tj. logaritmus o základu e (což je tzv. Eulerovo číslo, jehož hodnota je přibližně $2.718281828459\ldots$).

Poznámka: Toto tvrzení je využito např. při analýze časové složitosti algoritmu Quicksort v průměrném případě, která byla uvedena na přednášce.

Příklad 4: Navrhněte způsob, jak lze Turingův stroj s obecnou páskovou abecedou Γ simulovat Turingovým strojem s páskovou abecedou $\{0, 1, \square\}$.

Jak se při této simulaci změní počet provedených kroků?

Příklad 5: Navrhněte způsob, jak Turingův stroj se dvěma páskami simulovat pomocí Turingova stroje s jednou páskou.
Jaký je vztah mezi počtem kroků, které během výpočtu vykoná původní dvoupáskový stroj, a počtem kroků, které na stejném vstupu vykoná jednopáskový stroj sestrojený vámi navrženou konstrukcí?
Zobecněte navržený postup tak, aby fungoval pro stroje s libovolně velkým (ale konečným) počtem pásek — tj. pro stroje se třemi páskami, čtyřmi páskami, atd.

Příklad 6: Navrhněte a detailně popište způsob, jak může stroj RAM efektivně simulovat činnost Turingova stroje:

a) Uvažujte nejprve Turingův stroj s jednou páskou, která je jednostranně nekonečná.

b) Uvažujte Turingův stroj s jednou páskou, která je nekonečná na obě strany.

c) Uvažujte Turingův stroj s více páskami a s více hlavami na každé pásce, přičemž tyto pásky mohou být nekonečné na obě strany.

Příklad 7: Vysvětlete, co udělá univerzální Turingův stroj U, pokud dostane jako vstup slovo tvaru $\mathtt{u\#v}$, kde slovo $\mathtt{u}$ je kódem stroje U.