

Datové struktury I

1. přednáška: Úvod do amortizované složitosti

Jirka Fink

<https://ktiml.mff.cuni.cz/~fink/>

Katedra teoretické informatiky a matematické logiky
Matematicko-fyzikální fakulta
Univerzita Karlova v Praze

Zimní semestr 2024/25

Licence: Creative Commons BY-NC-SA 4.0

Kontakt

E-mail fink@ktiml.mff.cuni.cz

Homepage <https://ktiml.mff.cuni.cz/~fink/>

Konzultace Individuální domluva

Cíle předmětu

- Naučit se navrhovat a analyzovat netriviální datové struktury
- Porozumět jejich chování – jak asymptoticky, tak na reálném počítači
- Zajímá nás nejen chování v nejhorším případě, ale i průměrně/amortizovaně
- Nebudujeme obecnou teorii všech DS ani neprobíráme všechny varianty DS, ale ukazujeme na příkladech různé postupy a principy

- M. Mareš: Lecture notes on data structures, 2019.
- M. Mareš, T. Valla: Průvodce labyrintem algoritmů, CZ.NIC, 2017.
- A. Koubková, V. Koubek: Datové struktury I. MATFYZPRESS, Praha 2011.
- T. H. Cormen, C.E. Leiserson, R. L. Rivest, C. Stein: Introduction to Algorithms. MIT Press, 2009
- K. Mehlhorn: Data Structures and Algorithms I: Sorting and Searching. Springer-Verlag, Berlin, 1984
- D. P. Mehta, S. Sahni eds.: Handbook of Data Structures and Applications. Chapman & Hall/CRC, Computer and Information Series, 2005
- E. Demaine: Cache-Oblivious Algorithms and Data Structures. 2002.
- R. Pagh: Cuckoo Hashing for Undergraduates. Lecture note, 2006.
- M. Thorup: High Speed Hashing for Integers and Strings. Lecture notes, 2014.
- M. Thorup: String hashing for linear probing (Sections 5.1-5.4). In Proc. 20th SODA, 655-664, 2009.
- Záznamy z přednášek Martina Mareše a Petra Gregora (na SIsu)

Implementace algoritmů a datových struktur (NTIN106)

- Naučit studenty efektivně implementovat datové struktury v rozumném čase bez únavného hledání chyb
- Předmět je praktický — zápočet je za implementaci
- Rozvrh: Středa, 9:00, S10

Large Scale Optimization

- Postupy řešení algoritmicky těžkých optimalizačních úloh
- Dva předměty, které se v letním semestru střídají každé dva roky:
 - Exaktní metody (NOPT059)
 - Metaheuristiky (NOPT061)

Bakalářské a diplomové práce

- Přírodou inspirované algoritmy a kombinatorická optimalizace
- Problémy z teorie grafů a kombinatoriky

- 1 Amortizovaná složitost: binární čítač, dynamické pole
- 2 Stromy: $BB(\alpha)$ -stromy, splay stromy, (a,b) -stromy
- 3 Paměťová hierarchie: transpozice matic, merge sort
- 4 Hešování: volba hešovací funkce a řešení kolizí
- 5 Bloom filtry
- 6 Suffixové pole
- 7 Geometrické datové struktury: intervalové stromy
- 8 Paralelní datové struktury nepotřebující zámky

Jaké datové struktury znáte?

- Spojový seznam
- Pole
- Fronta, zásobník
- Vyhledávací stromy
- Hešovací tabulky
- Binární halda

Popis

- Paměť je rozdělená do bloků, které jsou indexovány čísly 1, 2, ...
- V každém bloku je uloženo přirozené číslo velikosti nejvýše $poly(n)$
- Význam hodnoty v bloku: klíč nebo hodnota prvku, adresa a další pomocné proměnné
- Instrukční sada: +, -, *, celočíselné dělení a modulo
- Argumenty instrukcí: konstanta, číslo bloku, blok obsahující číslo bloku

Složitost

- Paměťová: největší index použitého bloku
- Časová: počet vykonaných instrukcí

Motivace

- Uvažujme datovou strukturu, která zvládá nějakou operaci většinou velmi rychle
- Ale občas potřebuje reorganizovat svoji vnitřní strukturu, což operaci v těchto výjimečných případech značně zpomaluje
- Tudíž je časová složitost v nejhorším případě velmi špatná
- Představme si, že naše datová struktura je použita v nějakém algoritmu, který operaci zavolá mnohokrát
- V této situaci složitost algoritmu ovlivňuje celkový čas mnoha operací, nikoliv složitost operace v nejhorším případě
- Cíl: Chceme zjistit „průměrnou“ dobu výpočtu posloupnosti operací, případně celkovou složitost posloupnosti operací

Metody výpočtu amortizované složitosti

- Agregovaná analýza
- Účetní metoda
- Potenciální metoda

Binární čítač

- Máme n -bitový čítač inicializovaný libovolnou hodnotou
- Při operaci INCREMENT se poslední nulový bit změní na 1 a všechny následující jedničkové bity se změní na 0
- Počet změněných bitů v nejhorším případě je n
- Kolik bitů se změní při k operacích INCREMENT?

Agregovaná analýza

- Poslední bit se změní při každé operaci — tedy k -krát
- Předposlední bit se změní při každé druhé operaci — nejvýše $\lceil k/2 \rceil$ -krát
- i -tý bit od konce se změní každých 2^i operací — nejvýše $\lceil k/2^i \rceil$ -krát
- Celkový počet změn bitů je nejvýše
$$\sum_{i=0}^{n-1} \lceil k/2^i \rceil \leq \sum_{i=0}^{n-1} (1 + k/2^i) \leq n + k \sum_{i=0}^{n-1} 1/2^i \leq n + 2k$$
- Časová složitost k operací INCREMENT nad n -bitovým čítačem je $\mathcal{O}(n + k)$
- Jestliže $k = \Omega(n)$, pak amortizovaná složitost na jednu operaci je $\mathcal{O}(1)$

Účetní metoda

- Změna jednoho bitu stojí jeden žeton a na každou operaci dostaneme dva žetony
- Invariant: U každého jedničkového bitu si uschováme jeden žeton
- Při inkrementu máme vynulování jedničkových bitů předplaceno
- Oba žetony poskytnuté k vykonání operace využijeme na jedinou změnu nulového bitu na jedničku a předplacení jeho vynulování
- Na začátku potřebujeme dostat nejvýše n žetonů
- Celkově dostaneme na k operací $n + 2k$ žetonů
- Amortizovaný počet změněných bitů při jedné operaci je $\mathcal{O}(1)$ za předpokladu $k = \Omega(n)$

Potenciální metoda

- Potenciál nulového bitu je 0 a potenciál jedničkového bitu je 1
- Potenciál čítače je součet potenciálů všech bitů ①
- Potenciál po provedení j -té operace označme Φ_j
- Skutečný počet změněných bitů při j -té operaci označme T_j ②
- Chceme spočítat amortizovaný počet změněných bitů, který označíme A
- Pro každou operaci j musí platit $T_j \leq A + (\Phi_{j-1} - \Phi_j)$ pro libovolnou operaci j ③
- Zvolíme $A = 2$
- Celkový počet změněných bitů při k operacích je

→ skutečný čas *→ změna potenciálu*

$$\sum_{j=1}^k T_j \leq \sum_{j=1}^k (2 + \Phi_{j-1} - \Phi_j) \leq 2k + \Phi_0 - \Phi_k \leq 2k + n,$$

protože $0 \leq \Phi_j \leq n$ ④

- 1 V tomto triviálním příkladu je potenciál přesně počet žetonů v účetní metodě.
- 2 Φ_0 je potenciál před provedení první operace a Φ_k je potenciál po poslední operaci.
- 3 Toto je zásadní fakt amortizované analýzy. Potenciál je jako banka, do které můžeme uložit peníze (čas), jestliže operace byla levná (rychle provedená). Při drahých (dlouho trvajících) operacích musíme naopak z banky vybrat (snížit potenciál), abychom operaci zaplatili (stihli provést v amortizovaném čase). V amortizované analýze je cílem najít takovou potenciální funkci, že při rychle provedené operaci potenciál dostatečně vzroste a naopak při dlouho trvajících operacích potenciál neklesne příliš moc.
- 4 Součtu $\sum_{j=1}^k (\Phi_{j-1} - \Phi_j) = \Phi_0 - \Phi_k$ se říká teleskopická suma a tento nástroj budeme často používat.

Definice

Potenciál Φ je funkce, která každý stav datové struktury ohodnotí nezáporným reálným číslem. Operace nad datovou strukturou má amortizovanou složitost A , jestliže vykonání libovolné operace splňuje

$$T \leq A + (\Phi(S) - \Phi(S')),$$

kde T je skutečný čas nutný k vykonání operace, S je stav před jejím vykonáním a S' je stav po vykonání operace.

Příklad: Inkrementace binárního čítače

- Potenciál Φ je definován jako počet jedničkových bitů v čítači
- Skutečný čas T je počet změněných bitů při jedné operaci INCREMENT
- Amortizovaný čas A je 2
- Platí $T \leq A + (\Phi(S) - \Phi(S'))$

Zjednodušený zápis

$\Phi := \Phi(S)$ a $\Phi' := \Phi(S')$

Dynamické pole

- Máme zásobník (prvky přidáváme na konec a z konce i mažeme)
- Počet prvků označíme n a velikost pole p
- Jestliže $p = n$ a máme přidat další prvek, tak velikost pole zdvojnásobíme
- Jestliže $p = 4n$ a máme smazat prvek, tak velikost pole zmenšíme na polovinu

Intuitivní přístup ①

- Zkopírování celého pole trvá $\mathcal{O}(n)$
- Jestliže po realokaci pole máme n prvků, pak další realokace nastane nejdříve po $n/2$ operacích INSERT nebo DELETE ②
- Amortizovaná složitost je $\mathcal{O}(1)$

Agregovaná analýza: Celkový čas

- Necht k_i je počet operací mezi $(i - 1)$ a i -tou realokací $\Rightarrow \sum_i k_i = k$
- Při první realokaci se kopíruje nejvýše $n_0 + k_1$ prvků, kde n_0 je počáteční počet
- Při i -té realokaci se kopíruje nejvýše $2k_i$ prvků, kde $i \geq 2$ ③
- Celkový počet zkopírovaných prvků je nejvýše $n_0 + k_1 + \sum_{i \geq 2} 2k_i \leq n_0 + 2k$

- 1 V analýze počítáme pouze čas na realokaci pole. Všechny ostatní činnosti při operacích INSERT i DELETE trvají $\mathcal{O}(1)$ v nejhorším čase. Zajímá nás počet zkopírovaných prvků při realokaci, protože předpokládáme, že kopírování jednoho prvku trvá $\mathcal{O}(1)$.
- 2 Po realokaci a zkopírování je nové pole z poloviny plné. Musíme tedy přidat n prvků nebo smazat $n/2$ prvků, aby došlo k další realokaci.
- 3 Nejhorším případem je posloupnost INSERT, kdy zdvojnásobíme počet prvků, které poté musíme realokovat.

Potenciální metoda

- Uvažujme potenciál

$$\Phi = \begin{cases} 0 & \text{pokud } p = 2n \\ n & \text{pokud } p = n \\ n & \text{pokud } p = 4n \end{cases}$$

a tyto tři body rozšíříme po částech lineární funkcí

- Explicitně

$$\Phi = \begin{cases} 2n - p & \text{pokud } p \leq 2n \\ p/2 - n & \text{pokud } p \geq 2n \end{cases}$$

- Změna potenciálu při jedné operaci bez realokace je $\Phi' - \Phi \leq 2$ ①
- Skutečný počet zkopírovaných prvků T vždy splňuje $T + (\Phi' - \Phi) \leq 2$ ②
- Celkový počet zkopírovaných prvků při k operacích je nejvýše
 $2k + \Phi_0 - \Phi_k \leq 2k + n_0$
- Celkový čas k operací je $\mathcal{O}(n_0 + k)$
- Amortizovaný čas jedné operace je $\mathcal{O}(1)$

1

$$\Phi' - \Phi = \begin{cases} 2 & \text{pokud přidáváme a } p \leq 2n \\ -2 & \text{pokud mažeme a } p \leq 2n \\ -1 & \text{pokud přidáváme a } p \geq 2n \\ 1 & \text{pokud mažeme a } p \geq 2n \end{cases}$$

2 Jestliže nedojde k realokaci, pak $T = 0$.

Při realokaci musíme nejprve zkopírovat všechny prvky, kterých je $T = \Phi$, čímž potenciál klesne na 0. Poté přidáme/smažeme prvek, čímž potenciál vzroste nejvýše na $\Phi' \leq 2$.

Proč je potenciál nezáporný?

Naivní dynamizace pole

- Při vkládání prvku zvětšujeme pole jen o jedna
- $T = n + 1$: musíme zkopírovat $n + 1$ prvků
- Zvolme potenciál $\Phi(n) = -\binom{n}{2}$
- Platí $T \leq 1 + \Phi(n) - \Phi(n + 1)$
- Zvolíme $A = 1$ a vyšla nám konstantní amortizovaná složitost

Pro použití definice amortizované složitosti musí být potenciál musí být zdola omezený.

Celková složitost k operací

- Na začátku je pole prázdné, na konci má k prvků
- Celkový čas je

$$k \cdot A + \phi(0) - \phi(k) = k - 0 + \binom{k}{2} = \binom{k+1}{2} = \Theta(k^2)$$

Pokud počítáme celkový čas několika operací, tak je pokles potenciálu započítaný, takže nemusí být zdola omezený.

- Líně vyvažované stromy ($BB(\alpha)$ -stromy)
- Samovyvažovací stromy (Splay stromy)