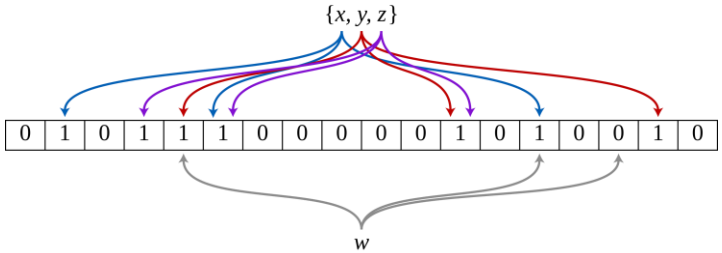
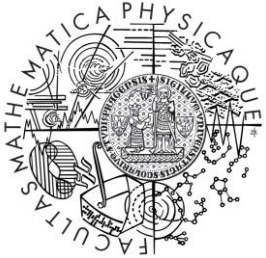
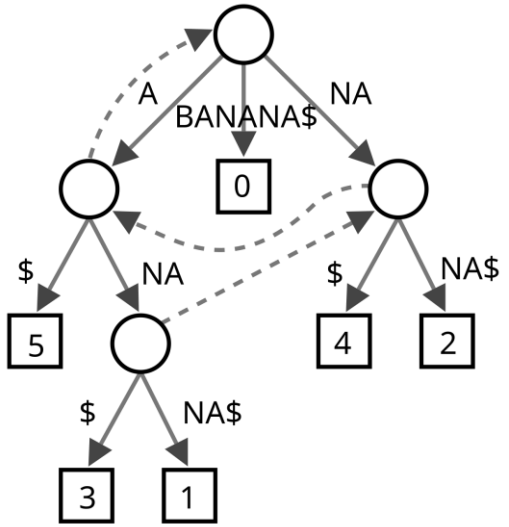
Datové struktury 1







ZS 25/26

Pavel Veselý

vesely+ds1@iuuk.mff.cuni.cz

https://iuuk.mff.cuni.cz/~vesely/vyuka/Podzim25/ds1.html

Cíle a předpoklady

- Naučit se lepší datové struktury
  - Rychlé, paměťově úsporné, adaptivní, dynamické
- Techniky návrhu a analýzy efektivních DS
- Analýzy DS nad rámec chování v nejhorším případě
- Umět navrhnout DS pro specifické úkoly
- Jak se chovají DS v praxi?

Předpokládané znalosti:

- Základní DS, např. binární vyhledávací stromy (BVS)
- Diskrétní matematika (kombinatorika, grafy)
- Základy teorie pravděpodobnosti (střední hodnota a její linearita, …)
- Programování

Konzultace – individuální domluva (co třeba pondělí 13-15 h)

Organizace předmětu

- Cvičení
- Zápočet za ≥**75 bodů z úkolů:**
  - Implementační: 7 úkolů po 10 bodech
    - např. doimplementovat operaci
    - Python/C++
  - Experimentální: 3 po 15 bodech
    - Připravené experimenty
  - Body upravuje cvičící
  - Termín 2 týdny po zadání (volnější pro posledních pár úkolů)
- Pravidla:
  - Musíte **rozumět všemu, co odevzdáte** – cvičící může ověřit při osobním setkání
  - **Text pište sami.** Citujte zdroje (přednášku nebo skripta nemusíte)
  - Řešení můžete diskutovat ústně, ale nesdílejte

Co je datová struktura?

Způsob uložení dat, který umožňuje určitou sadu operací

Dva pohledy:

1. "černá skříňka" s daným rozhraním, tedy operacemi
2. konkrétní implementace – pole, spojový seznam, binární vyhledávací strom, …

Příklady DS (rozhraní):

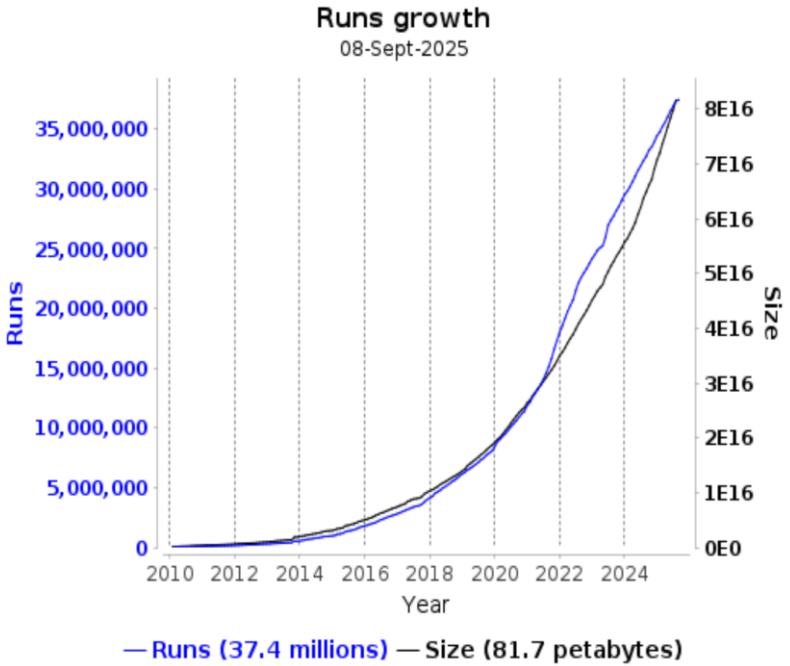
- fronta (*first in, first out*, FIFO) – operace `Enqueue` a `Dequeue`
- zasobník (*last in, first out*, LIFO) – operace `Push`, `Pop`
- prioritní fronta (implementace např. binární haldou)
- reprezentace množiny klíčů $X \subseteq U$, kde je U je univerzum
  - operace: `Insert, Delete, IsMember?`
- uspořádaná množina – navíc operace `Min, Max,` k`-týNejmenší, …`
- slovník – množina dvojic $(k; v)$, kde k je klíč a v hodnota

Dnešní výzvy v datových strukturách

European Nucleotide Archive

≈ 80 petabytů $= 8 \cdot 10^{16}$ bytů
DNA/RNA dat

Jak v tom vyhledávat?



Nedávný pokrok: [logan-search.org]

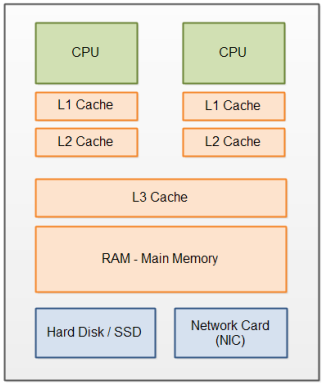
- vyhledávání v 50 PB v řádech minut

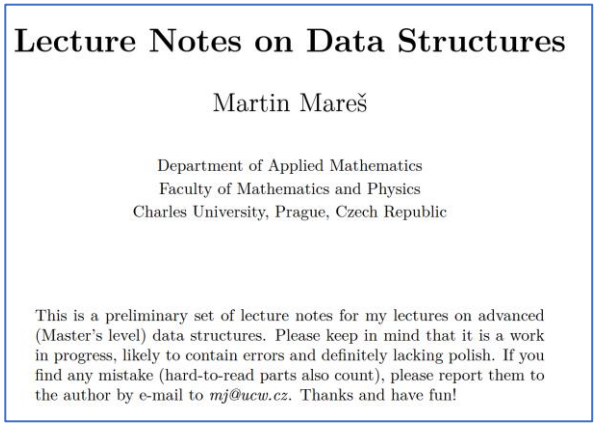
[Chikhi et al. 2024]

Založeno na pokročilém hešování:
Bloom filtry

Plán přednášky *Datové struktury 1*

1. **Amortizace** – lepší složitost v průměru přes hodně operací
2. Stromové DS:
   a. Líně vyvažované BVS (BB[α] stromy)
   b. Samovyvažovací (Splay stromy)
   c. (a, b)-stromy – amortizovaná analýza
3. Algoritmy pro **cache** – cache-aware vs. cache-oblivious
4. Hešování
   a. Silné hešovací funkce
   b. Lepší hešovací tabulky
   c. Bloom filtry
5. DS pro řetězce
6. DS pro geometrická data
7. Paralelní DS




Lecture Notes on Data Structures

Martin Mareš

Department of Applied Mathematics
Faculty of Mathematics and Physics
Charles University, Prague, Czech Republic

This is a preliminary set of lecture notes for my lectures on advanced (Master's level) data structures. Please keep in mind that it is a work in progress, likely to contain errors and definitely lacking polish. If you find any mistake (hard-to-read parts also count), please report them to the author by e-mail to *mj@ucw.cz*. Thanks and have fun!

Zdroje: https://iuuk.mff.cuni.cz/~vesely/vyuka/Podzim25/ds1.html

Navazující předměty

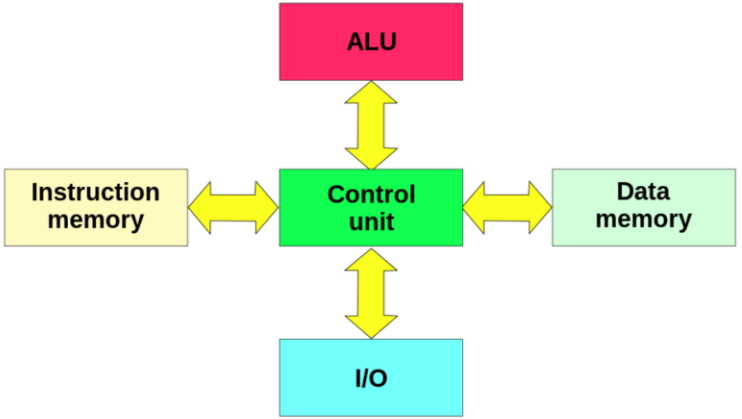
- **Datové struktury 2** (v LS, M. Mareš)
- Vybrané kapitoly z datových struktur (M. Mareš)
- Implementation of Algorithms and Data Structures (J. Fink)

Další související algoritmické přednášky

- Aproximační a online algoritmy (v LS, J. Sgall)
- Introduction to Parameterized Algorithms (M. Koutecký)
- Grafové algoritmy (M. Mareš)
- Vybrané kapitoly z výpočetní složitosti I: editační vzdálenost (M. Koucký)
- Proudové algoritmy pro velká data (P. Veselý)
- …

Výpočetní model – RAM (*Random Access Machine*)

- **Paměť**: neomezené pole adresované $\mathbb{Z}$
  - Obsahují celá čísla
- **Program**: posloupnost instrukcí
  - Aritmetické, logické
- Budeme používat **word-RAM**
  - Všechny operace trvají $O(1)$ času
  - Pouze s polynomiálně velká čísla, tedy s $k \cdot \log n$ bity
    - k je konstanta a n velikost vstupu



Zdroj: Nessa Ios, CC BY-SA 3.0, via Wikimedia Commons

Pro paralelní DS nebo analýzu chování ke cache procesoru zavedeme úpravy

Amortizace

- DS, která má většinu operací rychlých, třeba $O(1)$
  - Ale některé operace jsou pomalé! Třeba až $O(n)$
- Cíl: analýza pro m operací lepší než m krát nejhorší případ
  - Tedy „průměrná složitost přes velký počet operací“
- Př. 1: “Nafukovací pole“
  - Nový prvek přidáme na konec
  - Naplní se kapacita → realokujeme na dvojnásobnou kapacitu

Amortizace: binární počítadlo

- Pole P bitů

Algoritmus INC (zvýšení binárního počítadla o 1)

1. $i \leftarrow 0$
2. Dokud $P[i] = 1$:
3. $\quad P[i] \leftarrow 0$
4. $\quad i \leftarrow i + 1$
5. $P[i] \leftarrow 1$

Potenciálová metoda

Potenciál Φ = funkce, která přiřadí stavu DS hodnotu

- Typicky nezáporné
- Φ_i = potenciál po provedení i operací

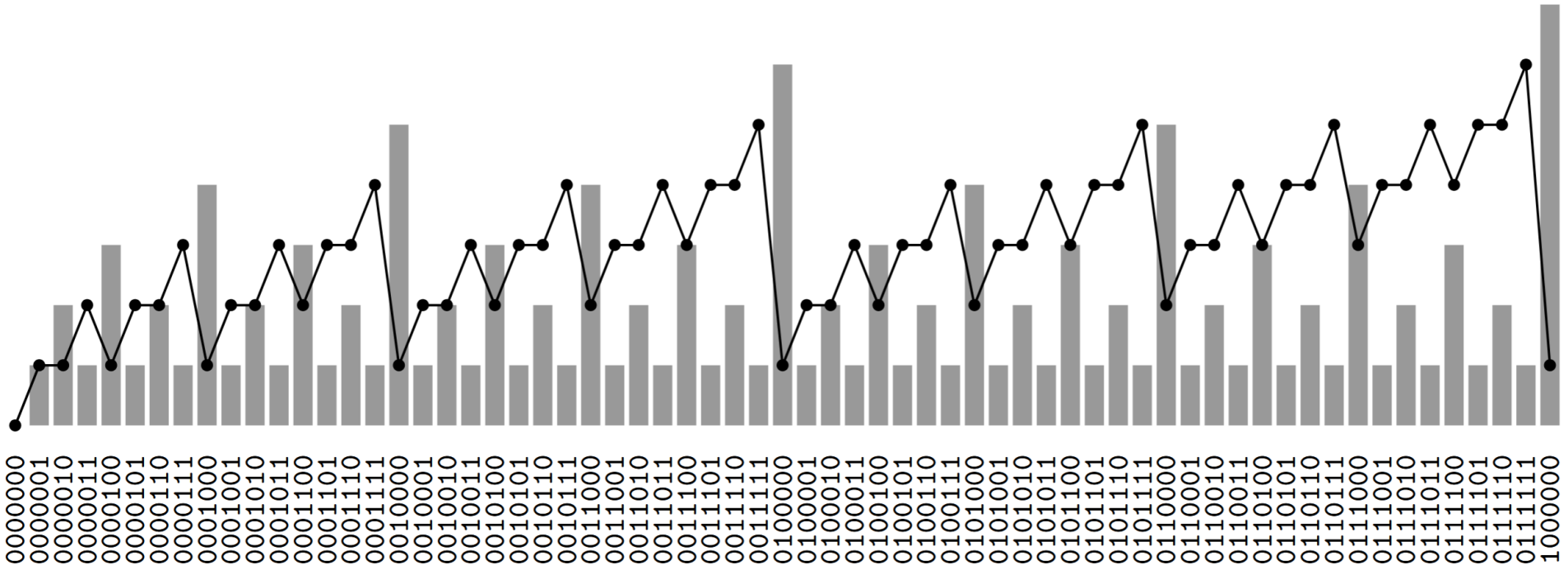
Def: Operace má amortizovanou složitost A, pokud existuje potenciál Φ t.ž. skutečná cena operace C splňuje:

$$C + \Phi_i - \Phi_{i-1} \leq A$$

Tedy:

- Drahou operací musí potenciál klesnout
- Každá operace zvedne potenciál jen omezeně

Potenciál pro binární počítadlo



Domácí úkol (dobrovolný)

- … jen pro občany ČR

Jděte volit!

Příště

- Zajímavější amortizace
- Binární vyhledávací stromy
  - Líně vyvažované přebudováním
  - Splay stromy: samovyvažovací