Cíle přednášky

- naučit se navrhovat a analyzovat netriviální DS
- porozumět jejich chování – jak asymptoticky, tak na reálném počítači
- zajímá nás nejen chování v nejhorším případě, ale i průměrné/amortizované
- nebudujeme obecnou teorii všech DS, spíše ukazujeme na příkladech různé principy

Povinnosti

- zkouška z probrané látky
- zápočet: – 4 domácí úkoly po 100 bodech, je třeba získat alespoň 320 b.
  - úkol = implementace DS + měření + zpráva s grafem
  - na vypracování 3–4 týdny, první úkol zadáme dnes → zadáváme centrálně, opravují cvičící
  - výrazně doporučujeme nízkoúrovňové jazyky; v nouzi lze použít i Javu nebo C#, jiné po domluvě s cvičícím
  - nepoužívejte knihovní/cizí kód v samotné implementaci DS → ani vector; představujte si RAM/ASM
  - programujte/pište samostatně! (diskutovat můžete)
  - implementaci můžete zvolit sami, ale musí být efektivní (asymptoticky aspoň jako ta z přednášky)
  - přesná pravidla na webu přednášky

Literatura – viz web (http://mj.ucw.cz/vyuka/ds1/)

Model počítače

- formálně to může být RAM (rozhodně ne Turingův stroj) [formalizace ASM]
- umíme adresovat: čísla („rozumně velká“), pole (pevně velké), ~~paměť~~ záznamy (pevného formátu)
  - na data se odkazujeme ukazateli (adresami v paměti)
- instrukce pracují v konst. čase
- paměť měříme v buňkách, které pojmou 1 číslo/ukazatel

(*) vsuvka o rozhraních DS

Amortizace

- pokus o postihnutí případů, kdy v posloupnosti operací jsou pomalé operace vzácné

* „Nafukovací“ pole – zpočátku fixní velikost (řekněme 1)
  – přidáváme na konec, kdykoli se pole naplní, realokujeme na 2x

reprezentace posloupnosti
chceme: Append(x)
At(i)

realokace trvá $\Theta(n)$, ale děje se „zřídkakdy“

– n × Append
⇓
realokace na velikostech $2^0, 2^1, \dots, 2^k$ ← $2^k < n \le 2^{k+1}$

celkem jimi trávíme čas $\Theta(2^0 + \dots + 2^k) = \Theta(2^{k+1}) = \Theta(n)$

⇒ $\Theta(1)$ na 1 Append AMORTIZOVANĚ (TBD exactly)

AGREGAČNÍ METODA (počítáme celkový čas)

Co kdybychom pole chtěli i zmenšovat? (ubírat prvky z konce)

- realokace při poklesu zaplnění pod polovinu kapacity nefunguje ($\Theta(n)$ dlouhodobě) ... problém: "skoro prázdné se stane skoro plným"
- lepší: realokujeme při poklesu pod čtvrtinu, vždy velikost ≥ 1

Věta: Posloupnost libovolných n operací Append a RemoveTail trvá $\Theta(n)$.

Dk: Posloupnost rozdělíme na bloky, blok končí realokací (kromě posledního)

ÚČETNÍ METODA

- v posl. bloku se nerealokuje $\to$ nezajímá nás
- v 1. bloku trávíme čas $\Theta(1)$ realokací velikosti 1 $\to$ ok
- ostatní bloky: - začaly realokací na kapacitu C, po ní blok obsahoval $C/2$ prvků
  - na konci realokace buď nahoru (máme $>C$ prvků) nebo dolů ($< C/4$ prvků)
  - $\Rightarrow$ realokace stála $\Theta(C)$, přibylo/ubylo $> C/4$ prvků $\Rightarrow > C/4$ operací $\Big\} \Rightarrow \Theta(1)$ na operaci ∎

$\hookrightarrow$ čas pomalých operací jsme "přeúčtovali" na ty rychlé.

intuice s hustotou struktury $\in [1/4, 1]$

* Binární počítadlo

- model: posloupnost bitů (zápis čísla), zpočátku nulová, opakujeme Inc
- ~~Inc ve w.c. stojí $\Theta(\ldots)$~~
- Inc stojí $\Theta(1 + \#1$ na konci čísla$)$, což je až $\Theta(\#$ bitů$)$
- ukážeme, že amortizovaně Inc trvá $\Theta(1)$
- Myšlenka: spoříme čas, 1\$ $\approx$ čas na zpracování 1 číslice
  - prohlásíme, že Inc trvá 2 peníze – 1. spotřebujeme, 2. "uložíme na horší časy"

    Invariant: $\#\$ = \#1$

$\boxed{\ldots\ldots\ 0\,1\,1\,1\,1}$

$\hookrightarrow$ přepisujeme na 0, platíme z kasičky

$\hookrightarrow$ přepíšeme na 1: 1\$ to stojí, 1\$ dáváme do kasičky

PENÍZKOVÁ METODA

Potenciálová metoda

Náš "účet" se chová jako nějaký potenciál ... budeme ho značit Φ

- označme c_i skutečnou cenu i-té operace

  a_i amort. —— " ——

  Φ_i potenciál ~~po i-té~~ po i-té operaci
- pak platí: $\Phi_0 = 0, \ \forall i\ \Phi_i \geq 0$

  $\Phi_i = \Phi_{i-1} + (a_i - c_i)$

  v minulém příkladu $\Phi = \#1$
- pojďme to otočit: zvolíme konkrétní Φ a z toho vypočteme $a_i = c_i + \underbrace{\Phi_i - \Phi_{i-1}}_{\Delta\Phi}$
- pro posloupnost operací pak platí:

$$\sum_{i=1}^{m} c_i = \sum_{i=1}^{m} (a_i - \Phi_i + \Phi_{i-1}) = \left(\sum_{i=1}^{m} a_i\right) + \Phi_0 - \Phi_m$$

pokud $\forall \Phi_i \geq 0$, $\Phi_0 = 0$, je tohle ≤ 0

$\Downarrow$

$\sum c_i \leq \sum a_i$

ale obecně stačí $\Phi_0 \leq \Phi_m$

<u>Shrnutí:</u> • obecně má platit $\sum_i c_i \le \sum_i a_i$ pro každou posloupnost operací začínající počátečním stavem struktury

• existuje více možností, jak korektně zvolit amort. složitosti (odpovídají různým potenciálům)

BB[1/2]-stromy (Líně vyvažované stromy)

↳ varianta vyhledávacích stromů s velice jednoduchým vyvažováním a $O(\log n)$ amort. na operaci

<u>Nápad:</u> Aby byl strom log. hluboký, nesmí se velikosti (# prvků) příliš lišit

... chceme poměr mezi 1:2 a 2:1, ale máme potíže s ∅ podstromy ⇒ def. trochu jinak:

<u>Df:</u> Mohutnost $m(v)$ vrcholu v := # potomků v (včetně) ← romou zavést i $T(v)$ a pak $m(v) := |T(v)|$

Strom je <u>v rovnováze</u> ≡ $\forall v$ vrchol, $\forall s$ syn v: $m(s) \le 2/3\, m(v)$.

<u>Lemma:</u> Strom na n vrcholech, který je v rovnováze, má hloubku $O(\log n)$. } Find má $O(\log n)$ W.C.

• <u>Udržování rovnováhy</u> (jen pro Insert; Delete je snadné cvičení)

- ve vrcholech si pamatujeme jejich $m(v)$
- Insert přidá list ⇒ zvýšíme $m(v)$ na cestě do kořene a kontrolujeme rovnováhu
- pokud pro nějaké v invariant neplatí, vybereme nejvyšší takové, vše pod ním rozebereme a postavíme znovu dokonale vyvážené ⇒ čas $\Theta(m(v))$

<u>Věta:</u> Amort. složitost operace Insert je $O(\log n)$.

<u>Dk:</u> Zavedeme potenciál: $\Phi := \sum_v \varphi(v)$

$$\varphi(v) := \begin{cases} |m(\ell(v)) - m(p(v))| & \text{je-li to aspoň 2} \\ 0 & \text{jinak} \end{cases}$$

☺ zvyšování $m(v)$ na cestě zvýší Φ o $O(\log n)$] pokud nedošlo k přebudování, vychází amort. čas $O(\log n)$

<u>Přebudování ve v:</u> BÚNO $m(\ell(v))$ bylo $> 2/3\, m(v)$ ... proto $m(r(v)) < 1/3\, m(v)$

takže $\varphi(v) \ge 1/3\, m(v)$

- toto přebudováním klesne na 0
- ostatním vrcholům se $\varphi(\cdot)$ nezvýší

} $\Delta\Phi \le -\frac{1}{3} m(v)$

↓

přebudování stojí $\Theta(m(v))$, takže se zaplatí z potenciálu

<u>Pozn.:</u> Vyvažování pomocí mohutností: 1972 Edward Reingold – BB[α] stromy, ty se ovšem vyvažovaly rotacemi

☺ Všimněme si, jak jsme zacházeli s konstantami