Omezte následující funkce:

$\sum_{i=1}^{n}(7i^3 - 30i + 16) \approx \sum(i^3)$, a o tom už víme, že $\in \Theta(n^4)$

$\sum_{i=1}^{n}\left(8\frac{i}{2^i}\right) \approx 8\cdot\sum\frac{i}{2^i}$, a o tom už víme, že $\in \Theta(1)$

$\sum_{i=1}^{n}(5^i + i^{100}) \approx \sum 5^i$, a o tom už víme, že $\in \Theta(5^n)$

DÚ [$\sum_{i=1}^{n}\left(\frac{1}{i^{1.1}}\right)$ omezení shora: $\sum_{i=1}^{n}\frac{1}{i^{1.1}} \le \sum_{i=1}^{n}\frac{1}{i^2} \le \int_0^n i^{-2}\,di = 1 + \int_1^n i^{-2}\,di =$

$= 1 + \left[-i^{-1}\right]_1^n = 1 + \left[-n^{-1} + 1^{-1}\right] = 2 - \frac{1}{n} \le 2$ ← tohle bude vždy nezáporné

Omezení zdola: protože sčítáme, stačí $\sum i^{-1.1} \ge 1$, tedy $g(n) = 1$, $c_1 = 1$, $c_2 = 2$, $f(n) \in \Theta(g(n))$

přednáška 5.3.2019

Amortizovaná složitost operace: Je-li $T(n)$ čas potřebný pro opakování n instrukcí, pak amortizovaná složitost jedné instrukce je $T(n)/n$. Dá se to spočítat třemi způsoby:

1) **Agregační metoda:** přímo vypočteme $T(n)$ pro všech n instrukcí. Pak amortizovaná složitost jedné instrukce je $T(n)/n$

2) **Účetní metoda:** provedení instrukce má cenu c_i. Chci odhadnout kredit $\hat{c}_i$ tak, aby
$$\sum_{i=1}^{n}\hat{c}_i \ge \sum_{i=1}^{n} c_i$$

3) **Potenciálová metoda:** Označme D_i stav po provedení i-té operace. Každému D_i je přiřazeno nezáporné číslo, tzv. potenciál $\Phi(D_i)$. Označme c_i jako skutečnou cenu přechodu od D_{i-1} k D_i. Pak amortizovaná cena $\hat{c}_i$ příslušná D_i je definovaná jako:
$$\hat{c}_i = c_i + \Phi(D_i) - \Phi(D_{i-1})$$
Pak platí:
$$\sum_{i=1}^{n}\hat{c}_i = \sum_{i=1}^{n}\left(c_i + \Phi(D_i) - \Phi(D_{i-1})\right) = \sum_{i=1}^{n} c_i + \Phi(D_n) - \Phi(D_0)$$
Když předpokládáme $\Phi(D_k) \ge \Phi(D_0)$, tak vyjde vztah pro účetní metodu.

Př:

```
Increment(A)
i = 0
while i < A.length ∧ A[i] = 1
    A[i] := 0
    i := i+1
if i < A.length
    A[i] := 1
```

Př: Euclid (a,b)
1. if b=0
2. return a
3. else return Euklid (b, $\overbrace{\text{a mod b}}^{\text{zbytek po dělení}}$)

- což je nerekurzivně
Euclid (a,b)
1. x[1] := a
2. y[1] := b, i=1
3. while y[i] != 0:
4. x[i+1] := y[i]
5. y[i+1] := x[i] mod y[i]
6. i++
7. return x[i]

Chci ukázat, že Euclid (a,b) ∈ (O(log b))

Lemma 1: Označme x_k a y_k dvojici čísel $x_k > y_k$ po k-tém rekurzivním volání. Pak $y_{k+2} < \frac{y_k}{2}$

Lemma 2: Je-li $a > b \geq 1$ a algoritmus Euklid(a,b) potřebuje k rekurzivních volání, pak $a \geq F(k+2)$ a $b \geq F(k+1)$, kde F(i) je i-tý člen Fibonacciho posloupnosti: $F(0)=0$, $F(1)=1$

$$F(n) = F(n-1) + F(n-2) \text{ pro } n \geq 2$$

Lemma 3: Euclid (F(k+2), F(k+1)) potřebuje k rekurzivních volání.

Lemma 4: Pro každé $n \geq 0$ platí $F(n+2) \geq \left(\frac{3}{2}\right)^n$.

Důkaz lemmatu 1:

$y_{k+2} < y_{k+1}$, protože to jsou zbytky při dělení a postupně se zmenšují

1) když už platí $y_{k+1} \leq \frac{y_k}{2}$, pak bude platit i $y_{k+2} < \frac{y_k}{2}$

2) když naopak $y_{k+1} > \frac{y_k}{2}$, využijeme toho, že $y_{k+1} = x_k \bmod y_k = x_k - q \cdot y_k$

$y_{k+2} = x_{k+1} - q' \cdot y_{k+1} = y_k - q' \cdot y_{k+1}$, kde $q' \geq 1$, protože jsme opravdu dělili

Z toho dostaneme $y_{k+2} = y_k - q' y_{k+1} \leq y_k - y_{k+1}$

A protože $y_{k+1} > \frac{y_k}{2}$, tak $y_{k+2} \leq y_k - y_{k+1} < y_k - \frac{y_k}{2} = \frac{y_k}{2}$, tedy $y_{k+2} < \frac{y_k}{2}$ ■

Důkaz lemmatu 2: indukcí podle počtu volání fce Euklid

1) základní krok: 1 volání (k=1)

$b \geq F(k+1) = F(2) = 1$ To nutně platí, protože jakmile b=0, už jsme skončili a nedělali další krok → taky to platí, protože to je v předpokladu

$a \geq F(k+2) = F(3) = 2$ Z předpokladu máme $a > b$, tedy $a \geq 2$.

2) indukční krok:

předpokládáme, že když Euklid (a', b') potřebuje k volání, tak $a' \geq F(k+2)$, $b' \geq F(k+1)$

Zvolím $b = a'$, $a = b \cdot q + b'$. Pak bude potřebovat $\underbrace{k+1}$ volání, $a > b$ bude splněno a přitom:

$$b = a' \geq F(k+2) = F((k+1)+1)$$

$$a = bq + b' \geq b + b' = F((k+1)+1) + F(k+1) = F((k+1)+2)$$

Ukázali jsme tedy: $b \geq F((k+1)+1)$, $a \geq F((k+1)+2)$ ■

Důkaz lemmatu 4: indukcí podle n

1) základní krok (n=0):

$F(0+2) = 1$
$(3/2)^n = (3/2)^0 = 1$ } $1 \geq 1$ ✓

(n=1) ← musím ověřit obě počáteční podmínky, protože F je definováno obecně až pro $n \geq 2$

$F(1+2) = 2$
$(3/2)^1 = 1.5$ } $2 \geq 1.5$ ✓

2) indukční krok:

Předpokládejme, že platí $F(n) \geq (3/2)^{n-2}$ a $F(n+1) \geq (3/2)^{n-1}$. Pak

$$F(n+2) = F(n+1) + F(n) \geq \left(\frac{3}{2}\right)^{n-2} + \left(\frac{3}{2}\right)^{n-1} = \left(\frac{3}{2}\right)^n \cdot \left(\left(\frac{3}{2}\right)^{-2} + \left(\frac{3}{2}\right)^{-1}\right) =$$

$$= \left(\frac{3}{2}\right)^n \left(\frac{4}{9} + \frac{2}{3}\right) = \left(\frac{3}{2}\right)^n \cdot \underbrace{\left(\frac{10}{9}\right)}_{>1} \geq \left(\frac{3}{2}\right)^n$$ ■

Věta: Algoritmus Euklid(a,b) vyžaduje $\Theta \lg(b)$ rekurzivních volání.

Důkaz: dolní odhad: Díky lemmatům 2 a 3 víme, že

horní odhad: Nejhorší případ je Euklid$(F(k+2), F(k+1))$, kde je potřeba k volání.

$$b = F(k+1) = F((k-1)+2) \underset{\text{lemma 4}}{\geq} \left(\frac{3}{2}\right)^{k-1} = \left(\frac{3}{2}\right)^{k} \cdot \left(\frac{2}{3}\right) \quad / \log_{\frac{3}{2}}$$

$\log b \geq k \cdot [\text{konstanta}]$

$k \geq c \cdot \log b \qquad \Rightarrow$ Euklid $\in O(\log b)$ ∎

Dokazování správnosti algoritmu

Variant: Variant je hodnota udaná přirozeným číslem, která se během algoritmu snižuje až k nejmenší možné hodnotě. Při dokazování chceme najít vhodný variant a dokázat, že se snižuje. Variant může být eventuelně nekonečný, pak chceme, aby přesáhl nějakou mez.

Př: U Euklidova alg je Variantem zbytek po dělení, který se nutně bude snižovat, dokud nebude nulový.

Pomocí variantu dokážeme, že algoritmus skončí.

Invariant (podmíněná správnost) je tvrzení, které:
- platí na začátku
- platí po prvním vykonání cyklu
- jestliže platilo před cyklem, bude platit i po něm
- po ukončení algoritmu zaručuje správnost řešení

Př: U Euklidova alg: "dvojice [a,b] má stejné společné dělitele jako dvojice [b, a%b]" (tedy mají i stejné gcd.

Pomocí invariantu dokážeme, že algoritmus najde správné řešení

Př: Algoritmus na hledání nejmenší kostry

1) (inicializace)

$K := \{\}$

$S := \{\{v\} \mid v \in V\}$ ← S jsou komponenty souvislosti (tedy na začátku izolované vrcholy)

2) (výběr hrany)

Dokud S není jednoprvková:

Vybereme hranu $e \in E \setminus K$ tak, že vede mezi dvěma různými množinami C_1 a C_2 ($\in S$) a aspoň v jedné z množin to je nejlevnější hrana vedoucí z ní.

3) (úpravy)

odstraním komponenty, které se spojily hranou

$K := K \cup \{e\}$ ← přidám novou komponentu z těch dvou

$S := (S \setminus \{C_1, C_2\}) \cup \{C_1 \cup C_2\}$ ← Spojím dvě komponenty

Vstup: Souvislý graf $G = (V, E)$ s oceněním hran $a: E \to \mathbb{Z}$

Výstup: hrany minimální kostry K

Tomuto algoritmu odpovídá: Jarník-Prim: hledá nejlevnější hranu z vrcholu

Kruskal: seřadí hrany, bere nejlevnější

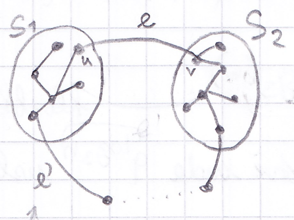
Variant: počet komponent v S, snižuje se z $|V|$ na 1
nebo počet hran v K, zvyšuje se z 0 na $n-1$

Invariant: K je vždy podmnožinou některé minimální kostry
- platí na začátku? Ano, prázdná množina je podmnožina čehokoliv
- platí na konci při správném řešení? Ano, K má stejně prvků jako kostra K' a zároveň $K \subseteq K'$, pak nutně $K = K'$, tedy K je kostra.

Chceme dokázat: konstruujeme-li podle schématu, pak $K \subseteq K'$ pro nějakou vhodnou K' pro každé přidání hrany:

Důkaz:

Indukční krok: Když $K \subseteq K'$ a vybereme hranu $e \in E \setminus K$ tak, že vede mezi S_1 a S_2 a pro S_1 je nejlevnější, pak $K_2 = K \cup \{e\}$.
1) Buď $e \in K'$, pak zjevně $K_2 \subseteq K'$.
2) Nebo $e \notin K'$.



může spojovat přímo S_1 a S_2, ale nemusí

$e \notin K'$, proto přidání další hrany uzavře kružnici, a na této kružnici nutně bude ležet hrana vycházející z S_1 – označme ji e'.

a je ocenění hran

Protože jsme vybrali e jako nejlevnější hranu, platí nerovnost $a(e) \leq a(e')$. Zároveň je K' nejlevnější kostra, $a(e') \leq a(e)$, proto $a(e) = a(e')$.

Zvolím kostru $K'' = K' \setminus \{e'\} \cup \{e\}$, pak $K_2 \subseteq K''$, a invariant platí. ∎

ceny e a e' jsou stejné, K'' je jiná minimální kostra

cvičení
6.3.2019

MT: $T(n) = a \cdot T(\frac{n}{b}) + f(n)$, $a, b \in \mathbb{N}$, $a \geq 1$, $b > 1$, $f(n)$ je nezáporná fce

1) jestliže $f(n) \in O(n^{\log_b a - \varepsilon})$ pro nějaké $\varepsilon > 0$, pak $T(n) \in \Theta(n^{\log_b a})$
2) —"— $f(n) \in \Theta(n^{\log_b a})$, pak $T(n) \in \Theta(n^{\log_b a} \cdot lg(n))$
3) —"— $f(n) \in \Omega(n^{\log_b a + \varepsilon})$ pro nějaké $\varepsilon > 0$ a jestliže $a \cdot f(\frac{n}{b}) \leq c \cdot f(n)$ pro nějaké $c < 1$ a $\forall n \geq n_0$, pak $T(n) \in \Theta(f(n))$

Př: $T(n) = 2T(\frac{n}{3}) + 1$, $T(1) = 1$ $\quad a = 2, b = 3, f(n) = 1$
Jsou splněny předpoklady MT/1, třeba pro $\varepsilon = 0.5$ rozhodně $1 \leq n^{\log_3 2 + 1}$
Z MT dostaneme: $T(n) \in \Theta(n^{\log_3 2}) \approx n^{0.63}$ ← číslo je jen pro představu, ne výsledek

Př: $T(n) = 9T(\frac{n}{3}) + n^2$ $\quad a = 9, b = 3, f(n) = n^2$
Je splněný předpoklad MT/2, zjevně $n^2 \in \Theta(n^{\log_3 9}) = \Theta(n^2)$
Z MT dostaneme: $T(n) \in \Theta(n^{\log_3 9} \cdot lg(n)) = \Theta(n^2 \cdot lg(n))$

Př: $T(n) = 30T(\frac{n}{25}) + n^{\frac{3}{2}} lg(n)$ $\quad a = 30, b = 25, f(n) = n^{\frac{3}{2}} \cdot lg(n)$
$\log_b a = \log_{25} 30$, bude to něco většího než 1, ale menšího než $\frac{3}{2}$
Chci zjistit, zda $f(n) \in \Omega(n^{\log_{25} 30 + \varepsilon})$
- jak porovnat $\log_{25} 30$ a $3/2$? ← přesněji
$\log_{25} x = 3/2$
$x = 25^{\frac{3}{2}} = \sqrt{25}^3 = 5^3 = 125$
- vidíme, že $f(n) \in \Omega(n^{\log_{25} 30 + \varepsilon})$ pro $\varepsilon \in (0, \frac{3}{2} - \log_{25} 30)$

násobení $lg(n)$ to nezmění, protože $lg(n) \geq 1$ pro $n \geq 2$

A druhá část předpokladů:
$30 \cdot f(\frac{n}{25}) \leq c\, f(n)$

$$30 \cdot \left(\frac{n}{25}\right)^{\frac{3}{2}} \cdot lg\left(\frac{n}{25}\right) = 30 \cdot n^{\frac{3}{2}} \cdot \left(\frac{1}{25}\right)^{\frac{3}{2}} \left(lg(n) - lg(25)\right) = \frac{30}{125} \cdot n^{\frac{3}{2}} \cdot \left(lg(n) - lg(25)\right) \leq$$
$$\leq \frac{6}{25} \cdot n^{\frac{3}{2}} lg(n)$$, druhý předpoklad platí pro $c = \frac{6}{25} < 1$

$T(n) \in \Theta(n^{\frac{3}{2}} \cdot lg(n))$

Př: $T(n) = 2T(\frac{n}{2}) + n\cdot lg(n)$ $\quad a=2,\ b=2,\ f(n) = n\, lg(n),\ \log_2 2 = 1$

Zkusíme použít MT/3

1) $f(n) \in \Omega(n^{1+\varepsilon})$ $\quad \lim\limits_{n\to\infty} \frac{n\cdot lg\, n}{n^{1+\varepsilon}} = \lim\limits_{n\to\infty} \frac{lg\, n}{n^{\varepsilon}} \overset{[L'H]}{=} \lim\limits_{n\to\infty} \frac{\frac{1}{n}\cdot\frac{1}{ln(2)}}{\varepsilon\, n^{(\varepsilon-1)}} = \lim\limits_{n\to\infty} \frac{1}{n\, ln(2)\cdot\varepsilon} = 0$

MT/3 nelze použít, musíme to udělat stromem

Př: $T(n) = 9T(\frac{n}{3}) + n^2,\ T(1) = 1$, řešme to stromem

0	n	n^2
1	$n/3$	$(\frac{n}{3^1})^2$
2	$n/9$	
⋮	⋮	$(1)^2$

n^2

$9^1\cdot(\frac{n}{3^1})^2 = n^2$

⋮

$9^k\cdot(\frac{n}{3^k})^2 = 9^k\cdot\frac{1}{9^k}\cdot n^2 = n^2$

$\frac{n}{3^k} = 1 \Rightarrow k = \log_3 n$, hloubka stromu je $\log_3(n)$

V každé větvi se strom větví 9x, v poslední vrstvě bude $9^{\overbrace{\log_3 n}^{\text{hloubka}}}$ uzlů

$9^{\log_3 n} = (3^2)^{\log_3 n} = (3^{\log_3 n})^2 = n^2$

$T(n) = \sum_{i=0}^{\log_3 n}(n^2) + n^2 = \log_3(n)\cdot n^2 + n^2 \Rightarrow T(n) \in \Theta(n^2\cdot lg(n))$

$f(n)$ v každé vrstvě (tady rovna není závislý na i)

počet uzlů v poslední vrstvě

Př: $T(n) = T(n-1) + n^3,\ T(1) = 1$

0	n	n	n^3
1	$n-1$	$n-1$	$(n-1)^3$
2	$n-2$	$n-2$	$(n-2)^3$
⋮		⋮	
		1	

$n - h = 1 \Rightarrow h = n-1$

$T(n) = \sum_{i=0}^{n-1}(n-i)^3 + 1 = \sum_{i=1}^{n} i^3 + 1$

$\sum_1^n i^3 + 1 \in \Theta(\sum_1^n i^3) = \underline{\underline{\Theta(n^4)}}$

tohle jsme počítali minule

Pokud bychom měli místo n^3 konstantu, tak řešením je $T(n) = \sum c + 1 = cn + 1$, $T(n) \in \Theta(n)$

Př: $T(n) = T(\sqrt{n}) + 1$, použijte substituci $T(2^m) = S(m)$

Chceme, aby $n = 2^m$, tedy $\sqrt{n} = 2^{m/2}$

Nová fce je $S(m) = S(m/2) + 1$ ← konstanta se nemění při substituci

Na $S(m)$ použijeme MT/2: $a=1,\ b=2,\ f(n)=1$

$\Rightarrow S(m) \in \Theta(\log m) = \Theta(lg\, m)$

Uděláme substituci zpátky $n = 2^m \Rightarrow m = lg(n)$

$T(n) \in \Theta(lg(lg(n)))$

Spočítejte amortizovanou složitost fce přidej(x), která přidává prvky do pole. Pokud je pole plné, udělá nové dvojnásobné pole, překopíruje prvky a pak teprve přidá x.

krok	velikost pole	počet operací
1	1	1 = 1 (zápis)
2	2	4=3+1 (3 je zvolená konstanta)
3	4	7=3·2+1
4	4	1
5	8	$3\cdot 2^2+1$
⋮		
n		$\dots 2^k \dots$

$T(n) = n + 3\cdot\underbrace{(1+2+2^2+\dots+2^{lg\, n})}_{k+1 \text{ členů}} = \quad \boxed{a_1\frac{q^n-1}{q-1}}$

konstanta

zápis x v každém kroku; násobky dvou, max počet je $lg(n) = k$

$= n + 3\cdot\frac{2^{k+1}-1}{2-1} = n + 3\cdot(2^{k+1}-1) \le n + 3\cdot(2n) = 7n$

$T(n) \in \Theta(n)$, amortizovaně $T(n)/n = 1$, tedy konstantní