

# Amortizirana časovna zahtevnost

Operacija ima amort. čas zahtevnost  $C$ ,  
če kakršno koli zaporedje  $n$  operacij zahteva  
večjemu  $C \cdot n$  enot časa

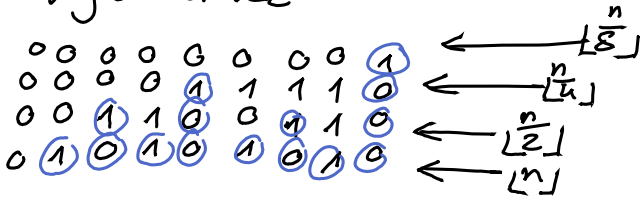
Primer: Dvojški števec predstavljen s tabelo  
ničel in enic  $B$ , ki predstavlja število  
 $\sum_i B[i] \cdot 2^i$

Za posamezno povečanje  $O(\log n)$  časa  
za  $n$  korakov potrebujemo  $O(n \log(n))$   
za en korak je amort. čas. zahtevnost  $O(\log(n))$

# Agregacijska metoda

$$C = \frac{\sum_{i=1}^n c_i}{n}$$

## 1. Dvojiški stevec



$$\sum c_i = n + \lfloor \frac{n}{2} \rfloor + \dots + \lfloor \frac{n}{2^{\log n}} \rfloor = \leq \sum_{k=0}^{\infty} \frac{n}{2^k} = 2n$$

$$\Rightarrow C = 2 \quad \text{Amortizirane zas. zah. je } O(1)$$

## 2. Dinamična tabela

Pr: vsakemu koraku vstavimo 1 element

Na koraku  $2^k + 1$  prestravimo se  $2^k$  el.

$$\begin{aligned} \sum c_i &= n + 1 + 2 + 4 + 8 + \dots + 2^{\lfloor \log n \rfloor} \leq \\ &\leq n + \sum_{i=0}^{\lfloor \log n \rfloor} 2^i = n + 2^{\lfloor \log n \rfloor + 1} - 1 \end{aligned}$$

$$\leq n + 2^{\log(n)+1} = 3n$$

## Računska metoda

Ideja: Operacije lažke „plačajo“ del stroška kasnejših operacij

### 1. Dvojisti števec

- Vsaka sprememba iz 0 na 1 plača 2 enoti.  
eno zase, eno za morebitno spremembo  $1 \rightarrow 0$
- Sprememba  $1 \rightarrow 0$  pokoristi privarčevana sredstva  
operacija bo plačala  $\hat{c}_i$  tako, da bo  $\sum c_i \leq \sum \hat{c}_i$

Na vsakem koraku plačamo najmanj 2 enoti.

Hkrat je ta dvojaj, saj vsaka sprememba  $1 \rightarrow 0$  sledi spremembi  $0 \rightarrow 1$

### 3. Unija z rangom

Glej PSA1

Dokažimo da je amortizirana čas. zaht.  $O(\log^* n)$

$$I_1 = [1, 1] \quad I_2 = [2, 2] \quad I_3 = [3, 4] \quad I_4 = [5, 16]$$

$$I_k = [2^k + 1, 2^{2^k}] = \{x \in \mathbb{N} \mid \log^* x = k\}$$

Y 26.2 manjhele



# Pretvorba statičnih podatkovnih struktur v dinamične

dinamična .... dodajamo in vzemamo elemente  
poldinamična .... samo eno ali drugo  
statična .... samo gledamo elemente

Elemente lahko shranimo v statično tabelo.  
ko odstranimo element, označimo v neki  
drugi tabeli vrednost False, da ga ne moremo  
več riniti

Recimo, da imamo P.S. nad element iz množice

$X$  s stopnjo  $S(X)$ , ki prorpia

- predpriprava  $l: P(X) \rightarrow S(X)$

- poizvedbe  $\varphi: S(X) \times Q \rightarrow A$

tako da poizvedbe lahko razpadejo

prek operacije  $\oplus: R \times R \rightarrow R$

da, je

$$\varphi(l(A_1 \cup A_2), g) = \varphi(l(A_1), g) \oplus \varphi(l(A_2), g)$$

Primeri:

- iskanje elementa v tabeli

$l$  .... uredi tabelo

$\varphi$  .... poišče z bisekcijo

$\oplus$  ....  $\vee$

- iskanje najbližje točke v ravnini v  $A^{\text{kon}} \in \mathbb{R}^k$

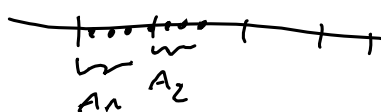
$l$  .... k-d drevo

$\varphi$  .... najbližja točka

$\oplus$  .... izbere izmed dveh točk bližje

Protiprimer

Število enotnih intervalov s krajšicami v  $A$

 ne razpade

Ideja je, da vodimo svoje  $L_0, L_1, \dots, L_k$

v  $L_i$  je  $2^i$  elementov, ki so razeni.

Nov element dodajemo na najnižji nivo

če je zaseden, jih preverimo skupaj in damo gor

Naj bo časovna zahtevnost  $L(A)$  enake  $P(n)$   $n = |A|$

• čas. zelt.  $P(L(A), g)$  je  $Q(n)$

• Naj bo  $P$  **superaditivna** ( $P(m+n) \geq P(m) + P(n)$ )

Tedaj je amortizirana čas. zelt. vstavljanja

enake 
$$P'(n) = \begin{cases} O(\underbrace{P(n)}) & P(n) \in \Theta(n^{1+\epsilon}) \quad \epsilon > 0 \\ P(n) O(\log n) & \text{v vseh primerih} \end{cases}$$

označuje zahtevnosti na novi strukturi

časovna zahtevnost proizvodnje

$$Q'(n) = \begin{cases} O(Q(n)) & ; Q(n) \in \Theta(n^\epsilon) \quad \epsilon > 0 \\ Q(n) O(\log n) & ; \text{v vseh primerih} \end{cases}$$

Dokaz:

$$1) \cdot Q'(n) = O(Q(n) \log(n)) \quad DN$$

$$2) \cdot Q'(n) = O(n^\epsilon) \quad \epsilon > 0 \Rightarrow Q'(n) = \Theta(Q(n)) \quad DN$$

$$3) \cdot P'(n)_{\text{amort}} = O\left(\frac{P(n) \log n}{n}\right)$$

$$4) \cdot P(n) = O(n^{1+\epsilon}) \quad \epsilon > 0 \Rightarrow$$

3) Recimo da smo ustavili  $n$  elementov

$$\dots \quad \text{Parabola} \quad \frac{n}{2} P(1) + \frac{n}{4} P(2) + \frac{n}{8} P(4) \dots + P\left(\frac{n}{2}\right)$$

Zaradi superaditivnosti

$$P\left(2 \cdot \frac{n}{2}\right) \geq 2 \cdot P\left(\frac{n}{2}\right)$$

$$\text{in podobno} \quad P(n) \geq 2^e P\left(\frac{n}{2^e}\right)$$

$$* \leq \frac{1}{2} (P(n) + P(n) + \dots + P(n)) \leq O(P(n) \log(n))$$

4) ker je  $P(n) = O(n^{1+\epsilon})$  je tudi  $O(n^{1+\epsilon})$

zato je skupaj čas. zaht.

$$\frac{1}{2} \sum_{i=0}^{\log n} \frac{n}{2^i} P(2^i) = \frac{n}{2} \sum_{i=0}^{\log n} \frac{O((2^i)^{1+\epsilon})}{2^i} =$$

$$O\left(\frac{n}{2} \sum_{i=0}^{\log n} (2^\epsilon)^i\right) = \frac{n}{2} O((2^\epsilon)^{\log n + 1}) =$$

$$= O(n^{\epsilon+1}) = O(P(n))$$

# Hitra fourierova transformacija

Imamo različne predstavitve p dinamov  
stopnje  $n$  z  $n+1=N$  podatki

| predstavitev                    | seštevanje | množenje | izračun $\sum_{j=0}^n a_j x_j^k$ |
|---------------------------------|------------|----------|----------------------------------|
| $a_0 + \dots + a_n x^n$         | $O(n)$     | $O(n^2)$ | $O(n)$                           |
| $a_n(x-x_0)\dots(x-x_{n-1})$    | ???        | $O(n)$   | $O(n)$                           |
| $(x_0, y_0), \dots, (x_n, y_n)$ | $O(n)$     | $O(n)$   | $O(n^2)$                         |

Poglejmo kako lahko za smiselno  
izbrane točke dobimo evalvacijo v času  
 $O(n \log n)$

Opazimo:

$$a_0 + a_1 x + \dots = a_0 + a_2 x^2 + a_4 x^4 + x(a_1 + a_3 x^2 + \dots)$$

$$\text{oz: } p(x) = p_s(x^2) + x p_l(x^2)$$

$$1) p(1) = p_s(1) + p_l(1)$$

$$2) p(-1) = p_s(1) - p_l(1)$$

rabimo samo  
2 vrednosti

$$3) p_s(-1) + i p_l(-1)$$

$$4) p_s(-1) - i p_l(-1)$$

$p_s$  razdelimo še na  $p_{ss} + p_{sl}$

Naj bo  $N = 2^k$  in  $\omega = e^{\frac{2\pi i}{N}}$

Vidimo da velja

$$p(\omega^k) = p_s(\omega^{2k}) + \omega^k p_l(\omega^{2k})$$

$$p(\omega^{k+\frac{N}{2}}) = p(-\omega^k) = p_s(\omega^{2k}) - \omega^k p_l(\omega^{2k})$$

$$\text{FFT}([a_0, a_1, \dots, a_{N-1}]) = [p(\omega^0), p(\omega^1), \dots, p(\omega^{N-1})]$$

$$\text{za } p(x) = a_0 + a_1 x + \dots + a_{N-1} x^{N-1}$$

časovna zahtevnost  $O(n \log n)$

Rabimo še inverz

Za dovolj velik  $N$  (DN) velja

$$\text{FFT}(\text{keet}(p \cdot g)) = \text{FFT}(\text{keet}(p)) \cdot \text{FFT}(\text{keet}(g))$$

če želimo  $\text{keet}(pg)$  rabimo inverz

$$\begin{bmatrix} p(\omega^0) \\ \vdots \\ p(\omega^{N-1}) \end{bmatrix} = \begin{bmatrix} 1 & 1 & \dots & 1 \\ \vdots & \omega & \dots & \omega^{N-1} \\ 1 & \omega^{N-1} & \dots & \omega^{N-1} \end{bmatrix} \begin{bmatrix} a_0 \\ \vdots \\ a_{N-1} \end{bmatrix}$$

$$[W_{ij}] = \omega^{ij}$$

$W$  je skoraj unitarna

$$[WW^H]_{ij} = \sum_{k=0}^{N-1} \omega^{ik} \underbrace{\bar{\omega}^{kj}}_{\omega^{-kj}} = \sum_{k=0}^{N-1} (\omega^{i-j})^k = \begin{cases} 0 & i \neq j \\ N & i = j \end{cases}$$

$$W^{-1} = \frac{1}{N} W^H$$

Želimo  $p$ , da bo  $\mathbb{Z}_p$  imel  $N$ -to koren enote  
 $p \equiv 1 \pmod{N}$  in  $g$  generata  $\mathbb{Z}_p$   
 je  $g^{\frac{p-1}{N}}$   $N$ -ti koren enote, saj je

$$\left(g^{\frac{p-1}{N}}\right)^N = g^{p-1} \equiv 1 \pmod{p} \quad (\text{meli Fermatov izrek})$$

Če so koeficienti večji o  $p$  lahko ravno  
 ustrezno večji;  $p$

še bolj pa uporabiti kitajski izrek o ostankih

2 niza  $s, t$   $\text{len}(s) = n$   
 $\text{len}(t) = m$

abeceda  $\Gamma$  kdaj je tangram  $s$

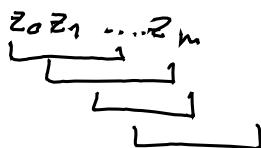
for  $x$  in  $s$   $T(h(x)) += 1$

for  $x$  in  $t$   $T(h(x)) -= 1$

check  $T = 0$

$$\textcircled{3} \quad h(u) = \sum_{i=0}^{|u|-1} u[i] \cdot 31^{|u|-1-i}$$

Ali ji  $s_0, s_1, \dots, s_n$  podniz  $z_0, \dots, z_m$   
 $\underbrace{\hspace{10em}}_{h(s)}$



preverjamo pod niz;

računamo bruiselno

$$\cancel{h_2} = \cancel{h_k} = 31.$$

$$h_{i+1} = 31(h_k - u_{k-1} \cdot 31^{n-1}) + u_{k+1} + 1$$

$10^4$  različnih izdelkov

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \in [0, 1]$$

$$k: \text{izdelki} \rightarrow \mathbb{N}_+$$

$$K = k_*(\{\text{izdelki}\})$$

$\pi$  permutacija  $\in S_K$

$$h(A) = \min_{a \in A} \pi(a)$$

$$\mathcal{P}(h(A) = h(B)) = J(A, B)$$

$$\mathcal{P}\left(\min_{a \in A} \pi(a) = \min_{b \in B} \pi(b)\right) = \mathcal{P}(x \in A \cap B) = \frac{|A \cap B|}{|A \cup B|}$$

$x := \min_{y \in A \cup B} \pi(y)$

$$\mathcal{P}(h(A) = h(B)) =$$

$$\mathcal{P}(\exists x \in A \cap B. \forall y \in \{\text{izdelki}\}. k(y) < k(x) \Rightarrow y \notin A \cup B) =$$

Fiksiramo  $x \in A \cap B \neq \emptyset$

$$= \mathcal{P}(y \notin A \cup B \mid k(y) < k(x))$$

???

V <sup>z</sup>agstörwne  
Isels

Za odprto nastavljanje potrebujemo močnejšo predpostavko enakomernega zgoščevanja:

Za vsako permutacijo  $\pi \in S_m$  je

$$P((h(x,0), h(x,1), \dots, h(x,m-1)) = \pi) = \frac{1}{m!}$$

Tej predpostavki ne zadosti nobena od prejšnjih funkcij:

- linearno in kvadratno zgoščevanje porodita le  $O(m)$  permutacij.

$$h(x,i)h'(x) + a \cdot i + b;^2$$

- dvojno zgoščevanje porodi  $O(m^2)$  permutacij, kar je v praksi dovolj dobro

$$h(x,i) = (h_1(x) + i h_2(x)) \bmod m$$

Trditev: Ob predpostavki enakomernega zgoščevanja je pričakovano število povzede:

- $\frac{1}{1-\alpha}$  pri neuspešnem iskanju
- $\frac{1}{\alpha} \ln\left(\frac{1}{1-\alpha}\right)$  pri uspešnem iskanju

$1 + \alpha + \alpha^2 + \dots$

# Univerzalne družine zgoščevalnih funkcij

Idealne zgoščevalne funkcije ni!

Ze  $\forall h$  lahko najdemo  $O(m)$  elementov, ki se shkejo v isto zgoščena vrednost.

Ideja je, da imamo družino funkcij  $\mathcal{H}$  in za vsako tabelo nekjuoro izberemo eno izmed njih

Definicija: Družina  $\mathcal{H}$  je univerzalna, če za vse  $x_1 \neq x_2 \in \mathcal{U}$  velja

$$|\{h \in \mathcal{H} ; h(x_1) = h(x_2)\}| \leq \frac{|\mathcal{H}|}{m}$$

$\uparrow$   
velikost tabele

Opomba:

- ob predpostavki univerzalne družine lahko „ponovimo“ iste dokaze kot pri predpostavki enostavnega enakomernega zgoščanja
- za odprto naslavljanje iščemo k-neodvisne družine

$\forall h \in \mathcal{H}. \rightarrow P(h(x_1) = y_1 \wedge \dots \wedge h(x_k) = y_k) = \frac{1}{m^k}$

• poimenovanje ni poendeno

Trditel: 2-neodvisna družina je 1-univerzalna

Dokaz:

$$\mathcal{P}(h(x_1) = h(x_2)) = \mathcal{P}\left(\bigvee_y h(x_1) = y \wedge h(x_2) = y\right)$$

$$\leq \sum_y \mathcal{P}(h(x_1) = y \wedge h(x_2) = y) \leq \sum_y \frac{1}{m^2} = \frac{1}{m}$$


Naj bo  $p > |U|$  praštevilo.  
Torej definiramo

$$\mathcal{H}_{p,m} = \{h_{a,b} ; a \in \mathbb{Z}_p^* \quad b \in \mathbb{Z}_p\}$$
$$h_{a,b}(x) = (a \cdot x + b \bmod p) \bmod m$$

Trditev:

$\mathcal{H}_{p,m}$  je univerzalna.

Dokaz:

Vzemimo  $x \neq y$ , daj  $h_{a,b}(x) = h_{a,b}(y)$

Označimo  $r = ax + b \bmod p$

$s = ay + b \bmod p$

Torej je  $r - s \equiv a(x - y) \bmod p$

ker je  $a \in \mathbb{Z}_p^*$  je tuj s  $p$ , ker je  $p > |U|$

je  $x - y$  tuj s  $p$  zato je

$r - s$  tuje  $p$  in zato  $r \neq s$

Torej trk nestane  $\Leftrightarrow$  ko je  $r \equiv s \bmod m$ .

Še več: Za fiksne  $(x, y)$  imamo bijekcijo  
med  $(a, b) \in \mathbb{Z}_p^* \times \mathbb{Z}_p$   $(r, s) \in (\mathbb{Z}_p \times \mathbb{Z}_p) - \Delta$ .

Torej je vrjetnost trka pri naključno izbrani  
 $(a, b)$  ista kot vrjetnost trka pri naključno  
izbranih  $(r, s)$

Zadani  $r$  so  $r$  trku

$$s \in \{r + km ; k \in \mathbb{N}\} \quad |\mathbb{Z}| = \frac{p}{m}$$

Torej je vrjetnost trka  $\frac{p}{m} \cdot \frac{1}{p} = \frac{1}{m}$

**Popolno zgoščevanje** - lahko zgradimo  
zgoščeno tabelo brez trkov s poizvedbami  
v času  $O(1)$

Ideja: če je  $m \geq n^2$ , je pričakovano število  
trkov  $\leq \frac{1}{2}$

Po neenakosti Markova  $P(x \geq t) \leq \frac{E(x)}{t}$   
je vrje trost trka  $< \frac{1}{2}$

$$E(\text{št trkov}) = E\left(\sum_{x \neq y} \begin{cases} 1; & h(x) = h(y) \\ 0; & \text{sicer} \end{cases}\right) =$$

$$= \frac{n(n-1)}{2} \cdot \frac{1}{m} < \frac{n^2}{2m}$$

Da porabimo manj kot  $O(n^2)$  prostora  
naredimo strukturo v dveh nivojih:

1. tabela  $T$  velikosti  $m$  z zgoščevalno  
funkcijo  $h \in \mathcal{H}^{\text{univ.}}$
2. Vse elemente, ki zasedajo mesto  $j$ ,  
shranimo v tabelo  $T_j$  velikosti  $n_j^2$

izberemo si velikejšo sekundarno  
izdel de so  $n_j^2$   
Torej

$$E\left(\sum_{j=1}^n n_j^2\right) = O(n)$$

Dokaz

$$\begin{aligned} E\left(\sum n_j^2\right) &= E\left(\sum n_j + 2 \binom{n_j}{2}\right) = \\ &= E\left(\sum n_j\right) + 2 E\left(\sum \binom{n_j}{2}\right) = \end{aligned}$$

$\sim$  št. trkov na prvem  
nivoju

$$= n + 2 \cdot E\left(\sum_{ij} I(\text{trk med } i, j)\right) =$$

$$\leq n + 2 \cdot \frac{1}{n} m \binom{n}{2} = 2n-1 \cdot O(n)$$

$\sim$  št. možnih  
trkov  
 $\sim$  vrjetost  
trka

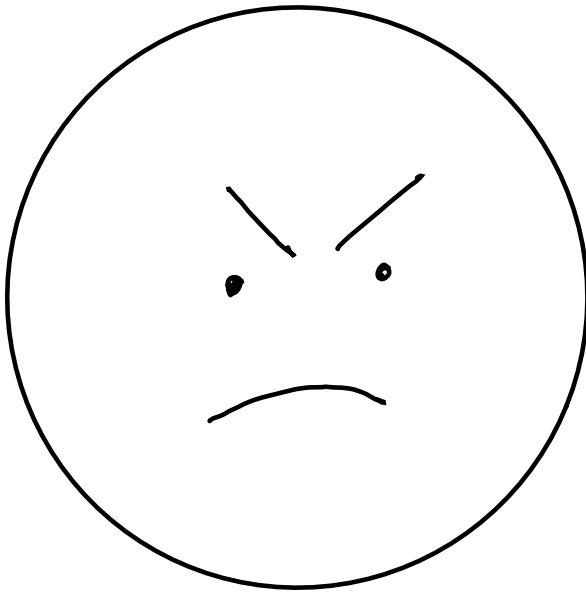
~~Drgacen način~~

~~$$\sum E\left(\binom{n_j}{2}\right) = \sum E\left(\left(\sum I(n(x_i)=j)\right)^2\right)$$~~

Opomba  $m=n$  je kompromis med  
velikostjo primerne tabele in  
skupno velikostjo sekundarne tabele

Wpjetnost da velikejš

tabele preseže  $4n < \frac{1}{2}$



## Bloomov filter:

Bloomov filter je PS ki podpira operaciji:

- dodaj ( $x$ ) ki doda  $x$  v PS in
- preveri ( $x$ ), ki za  $x$  vrne vredno resno DA, če je neki in največje NE, če ni neke.

Implementacije:

Imena:  $labelCB$  z  $m$  biti in k neodvisnih zgostovalnih funkcij,  $h: U \rightarrow [m]$

Element  $x$  vstane label, da in

$$B[h; x] := 1$$

## Preskoči seznam:

$0 \rightarrow 2 \rightarrow 3 \rightarrow 5 \rightarrow 8 \rightarrow 12 \rightarrow 21 \rightarrow 34 \rightarrow$   
 $0 \rightarrow 2 \longrightarrow 5 \rightarrow 8 \longrightarrow 21 \longrightarrow$   
 $0 \longrightarrow 5 \longrightarrow 21 \longrightarrow$

ke v istem času mečemo kovanec,  
da je približno  $\frac{1}{2}$  uspešni, da dobimo  
se bližjico

Vida, Vida, tralala XD,  
grdo me gleda, ker ne zna 

Trditev: Naj bo  $N$  število elementov v

Torej je:

- prikazna števila nivojev  $O(\log n)$
- pri čem. vztraje  $O(n) = 2^n + O(\log n)$
- prikazna dolžina pri iskanju  $O(\log n)$

Dokaz:

Verjetnost da se element pojavi na nivoju  $L_i = \frac{1}{2^i}$   $E(|L_i|) = \frac{n}{2^i}$

Očitno je  $[L_i \neq \emptyset] \leq |L_i|$

$\uparrow$   $\kappa$  karakteristična funkcija

Izračunamo število nivojev

$E(\sum [L_i \neq \emptyset]) = \sum_{i=0}^{\log n} E([L_i \neq \emptyset]) + \sum_{i=\log n+1}^{\infty} E([L_i \neq \emptyset])$

$$E(\sum [L_i \neq \emptyset]) = \sum_{i=0}^{\log n} E([L_i \neq \emptyset]) + \sum_{i=\log n+1}^{\infty} E([L_i \neq \emptyset])$$

$$\leq \log n \cdot 1 + \sum_{i=\log n+1}^{\infty} \dots \leq \log n + \underbrace{\sum_{i=2}^{\infty} \frac{n}{2^i}}_{\leq 2} \in O(\log n)$$

At neta 2.4 uvidimo je

$$\sum_{i=0}^{\log n} E(L_i) = \sum_{i=0}^{\log n} \frac{n}{2^i} = 2^n$$

Nałajman: ~~is~~ the reverse

treap (tree + heap)

|| Katerina's val demand  
queue + priority

# Funkcijske podatkovne strukture

Pozor: bomo na dve korente:

- Nespremenljivost P.S. in
- lenost izvajenja

## Nespremenljivost:

Da se dej smo delali s spremljivimi:

podatkovni in. strukturi (minjive/ephemeral)

V funkcijem programiraju si želeli  
referenčne transparentnosti: Vsaka vrednost je

ovrta svoji definiciji

želimo nespremenljivost (trajnih/persistent)

pod. strukturi

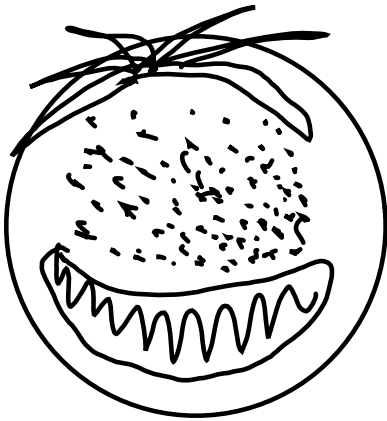
$r \leftarrow \text{ref } 0$

Primer:  $f_x = .x + x$

$g_x = \text{incr } r; r + x$

Ali:  $f(g_x) = g_x + g_x$  ?

Negativen, i.e. das ~~mo~~ s. folgendes  
 soemfivik peller, ~~am~~ u. d. h. d. h.  
 pa z. d. f. n. j. m. n. e. m. a. j. v. i. h. p. e. b. l. e. n.



Pameran ča ča ča

lačimo med

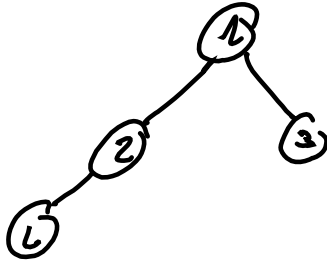
- abstraktnim, podallevnim, skulitram;  
 specifik a operativni n. h. v. i. h.  
 h. d. n. d. i. h. d. l. a. d. i. v. i. s. t. e. s.  
 p. d. z. t. j. e. . . . )
- implemetarizem; t. e. l. d. h. v. i. h. p. . . . s.  
 (sklad z v. i. z. m. s. e. m. m. a. n. i.)  
 o. h. t. a. b. e. l. e. - t. e. d. e. c

## Lerizastla hejico

levo nahljenjene hejico

- dvojide draveze
- lastnost min-hepice
- rang levega otoka je večji od rang desnega  
← dolžina desnega roba
- rekurencna z otok

Primo:



Trditiv:

Rang drveca  $\leq \lfloor \log_2(n+1) \rfloor$

Dobro:  $\mathcal{D}N$

Kajna presliza na kralj kralj  
je zbiranje  
Sed je vse same zbiranje

Dokaz:

Recimo da je to res za vse  $\{N < n\}$

$$\text{rob}(\text{dravo}) = \text{rob}(\text{desno}) + 1$$

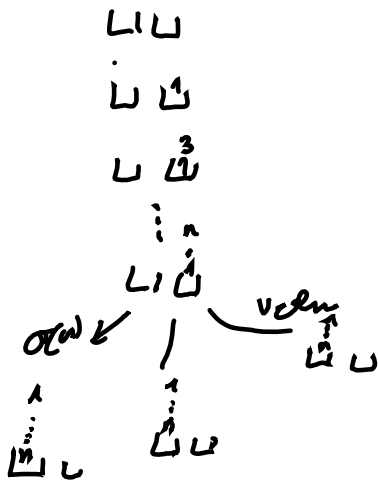
$$\text{rob}(\text{desno}) \leq \log(m+1)$$

$$\Rightarrow \text{rob}(\text{dravo}) \leq \log(m+1) + 1 =$$

$$\log((m+1) \cdot 2) = \log(2m + 1 + 1) \leq \log(n+1)$$

$$\uparrow$$
$$2m+1 \leq n$$

- amortizacija v primeru nespremenljivih podskupinskih struktur vodi v težave, saj lahko druge operacije izvedemo večkrat na istem stanju

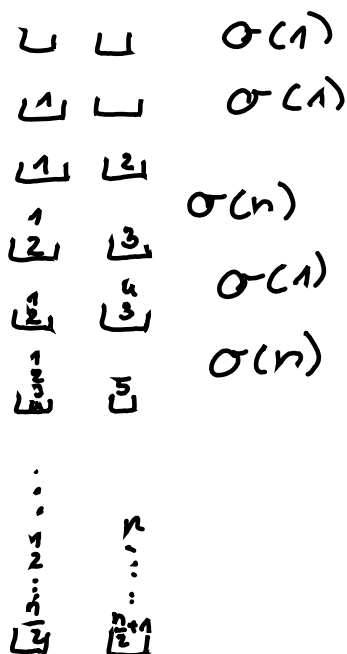


že  $n$  dedekinj in  $m$  odvečanij  $L$  pričakovali  $O(n+m)$  časa, dedekinj čas:  $O(n+mn)$

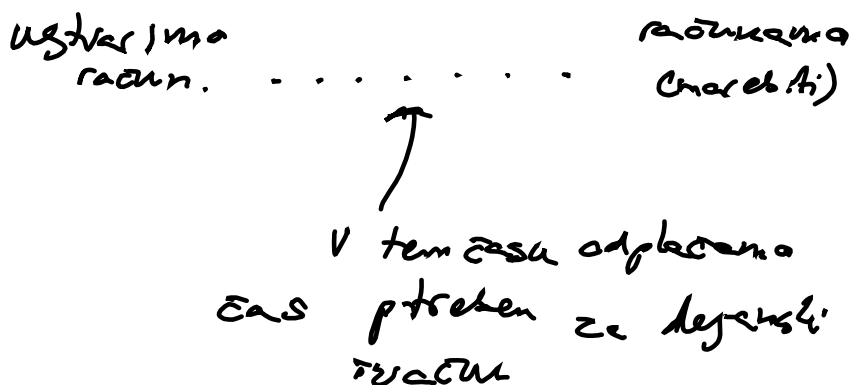
Rešitev: leno izvajanje: vse se skupaj na konec

Kako utemeljimo časovno zahtevnost

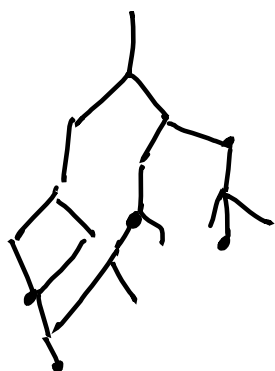
Leno izvajanje:



Ideja: Pri neučinkovitem izvajanju se račun izvede takoj, ko ga potrebujemo. Pri lenem izvajanju sta ti dve točki ločeni v času.



ko ustvarimo račun, si zahtevano dolgo, ki ga znamo z amortiziranimi operacijami, odložiti, preden se izvede



Formelno definiramo sled izvajanja kot graf v katerem so vozlišča izvedene operacije, povezave pa odvisnosti med njimi.

$i$  je preteklost operacije  $N$ , torej vsi predhodniki grafa

Vsakemu vozlišču  $v$  moramo dodeliti.

- Množico potencialnih stroškov  $S(v)$
- Množico odplačil  $a(v)$
- množico izvedenih stroškov  $r(v)$