Haldy

kuko

6.10.2020

Vybrané partie z dátových štruktúr

	pole	binárna halda
insert	$O(1)$	$O(\log n)$
extract-min	$O(n)$	$O(\log n)$
decrease-key	$O(1)$	$O(\log n)$

	pole	binárna halda
insert	$O(1)$	$O(\log n)$
extract-min	$O(n)$	$O(\log n)$
decrease-key	$O(1)$	$O(\log n)$
Dijkstra	$O(n^2)$	$O(m \log n)$

$$T_{Dijkstra} = n \times (\text{insert} + \text{delete min}) + m \times \text{decrease key}$$

	pole	binárna halda	d-árna halda
insert	$O(1)$	$O(\log n)$	$O(\log_d n)$
extract-min	$O(n)$	$O(\log n)$	$O(d \log_d n)$
decrease-key	$O(1)$	$O(\log n)$	$O(\log_d n)$
Dijkstra	$O(n^2)$	$O(m \log n)$	$O(m \log_d n + nd \log_d n)$

$$T_{Dijkstra} = n \times (\text{insert} + \text{delete min}) + m \times \text{decrease key}$$

	pole	binárna halda	d-árna halda
insert	$O(1)$	$O(\log n)$	$O(\log_d n)$
extract-min	$O(n)$	$O(\log n)$	$O(d \log_d n)$
decrease-key	$O(1)$	$O(\log n)$	$O(\log_d n)$
Dijkstra	$O(n^2)$	$O(m \log n)$	$O(m \log_{m/n} n)$

$$T_{Dijkstra} = n \times (\text{insert} + \text{delete min}) + m \times \text{decrease key}$$

ak $m = \Omega(n^{1+\varepsilon})$, tak $\log_{m/n} n = \log_{n^\varepsilon} n = O(1/\varepsilon)$

	pole	binárna halda	d-árna halda
insert	$O(1)$	$O(\log n)$	$O(\log_d n)$
extract-min	$O(n)$	$O(\log n)$	$O(d \log_d n)$
decrease-key	$O(1)$	$O(\log n)$	$O(\log_d n)$
Dijkstra	$O(n^2)$	$O(m \log n)$	$O(m \log_{m/n} n)$

$$T_{Dijkstra} = n \times (\text{insert} + \text{delete min}) + m \times \text{decrease key}$$

ak $m = \Omega(n^{1+\varepsilon})$, tak $\log_{m/n} n = \log_{n^\varepsilon} n = O(1/\varepsilon)$

	pole	d-árna halda	???
insert	$O(1)$	$O(\log_d n)$	$O(1)$
extract-min	$O(n)$	$O(d \log_d n)$	$O(1)$
decrease-key	$O(1)$	$O(\log_d n)$	$O(\log n)$
Dijkstra	$O(n^2)$	$O(m \log_{m/n} n)$	

$$T_{Dijkstra} = n \times (\text{insert} + \text{delete min}) + m \times \text{decrease key}$$

	pole	d-árna halda	???
insert	$O(1)$	$O(\log_d n)$	$O(1)$
extract-min	$O(n)$	$O(d \log_d n)$	$O(\log n)$
decrease-key	$O(1)$	$O(\log_d n)$	$O(1)$
Dijkstra	$O(n^2)$	$O(m \log_{m/n} n)$	

$$T_{Dijkstra} = n \times (\text{insert} + \text{delete min}) + m \times \text{decrease key}$$

	pole	d-árna halda	Fibonacciho halda
insert	$O(1)$	$O(\log_d n)$	$O(1)$
extract-min	$O(n)$	$O(d \log_d n)$	$O(\log n)$
decrease-key	$O(1)$	$O(\log_d n)$	$O(1)$
Dijkstra	$O(n^2)$	$O(m \log_{m/n} n)$	

$$T_{Dijkstra} = n \times (\text{insert} + \text{delete min}) + m \times \text{decrease key}$$

	pole	d-árna halda	Fibonacciho halda
insert	$O(1)$	$O(\log_d n)$	$O(1)$
extract-min	$O(n)$	$O(d \log_d n)$	$O(\log n)$
decrease-key	$O(1)$	$O(\log_d n)$	$O(1)$
Dijkstra	$O(n^2)$	$O(m \log_{m/n} n)$	$O(m + n \log n)$

$$T_{Dijkstra} = n \times (\text{insert} + \text{delete min}) + m \times \text{decrease key}$$

	pole	d-árna halda	Fibonacciho halda
insert	$O(1)$	$O(\log_d n)$	$O(1)$
merge			$O(1)$
extract-min	$O(n)$	$O(d \log_d n)$	$O(\log n)$
decrease-key	$O(1)$	$O(\log_d n)$	$O(1)$
Dijkstra	$O(n^2)$	$O(m \log_{m/n} n)$	$O(m + n \log n)$

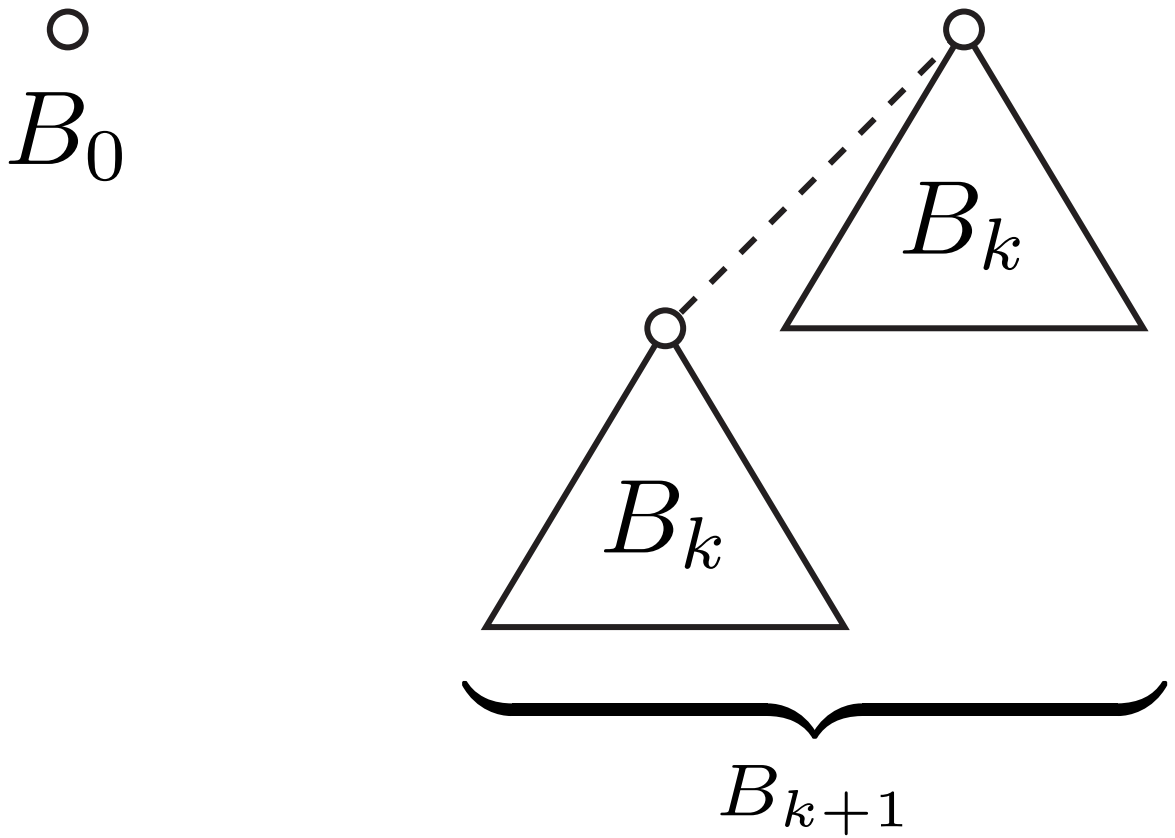
$$T_{Dijkstra} = n \times (\text{insert} + \text{delete min}) + m \times \text{decrease key}$$

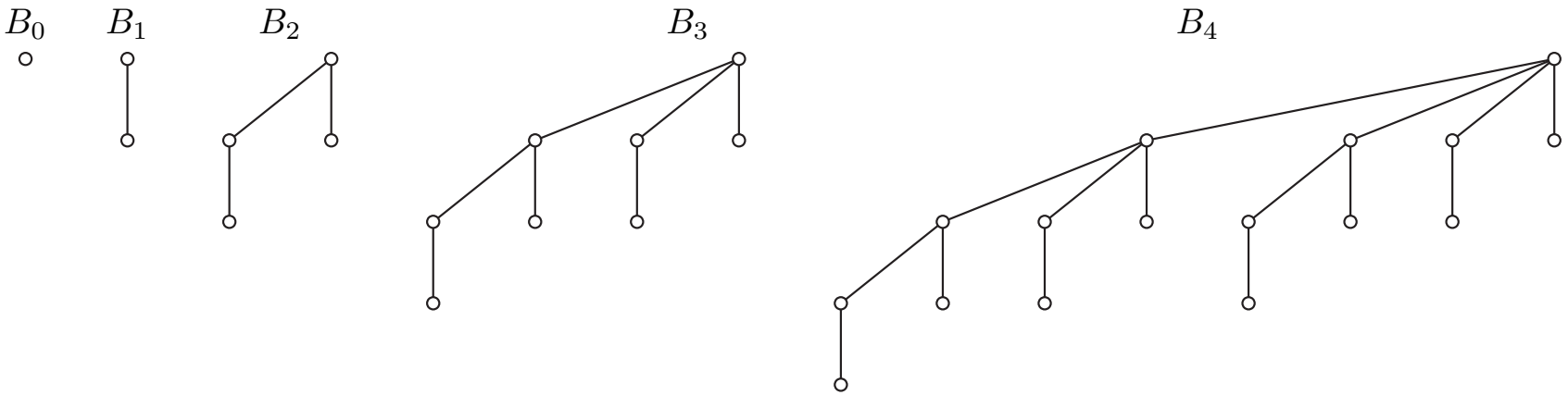
Binomiálna halda

kuko

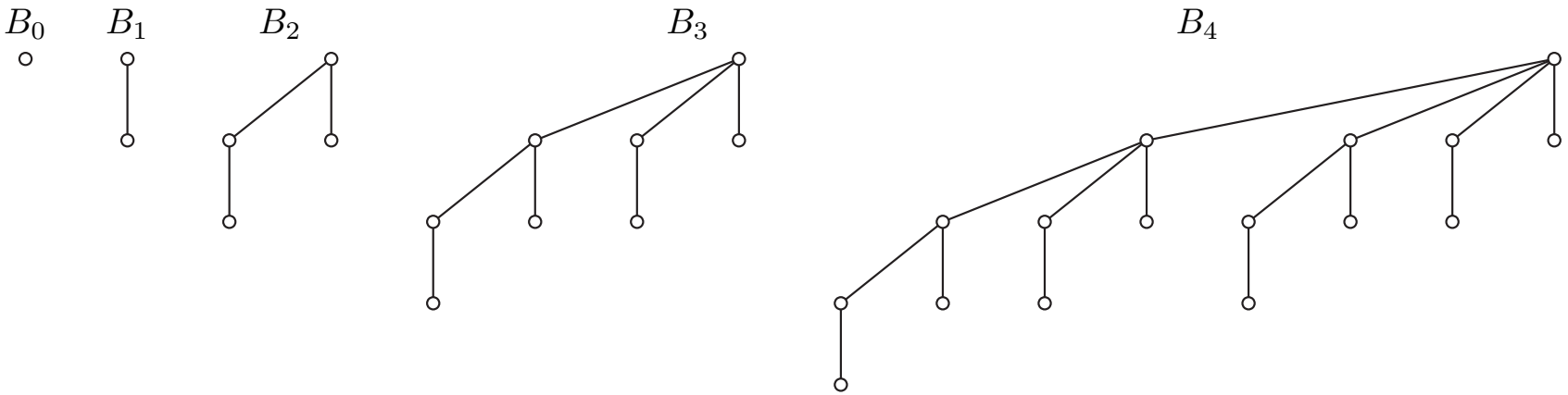
6.10.2020

Vybrané partie z dátových štruktúr

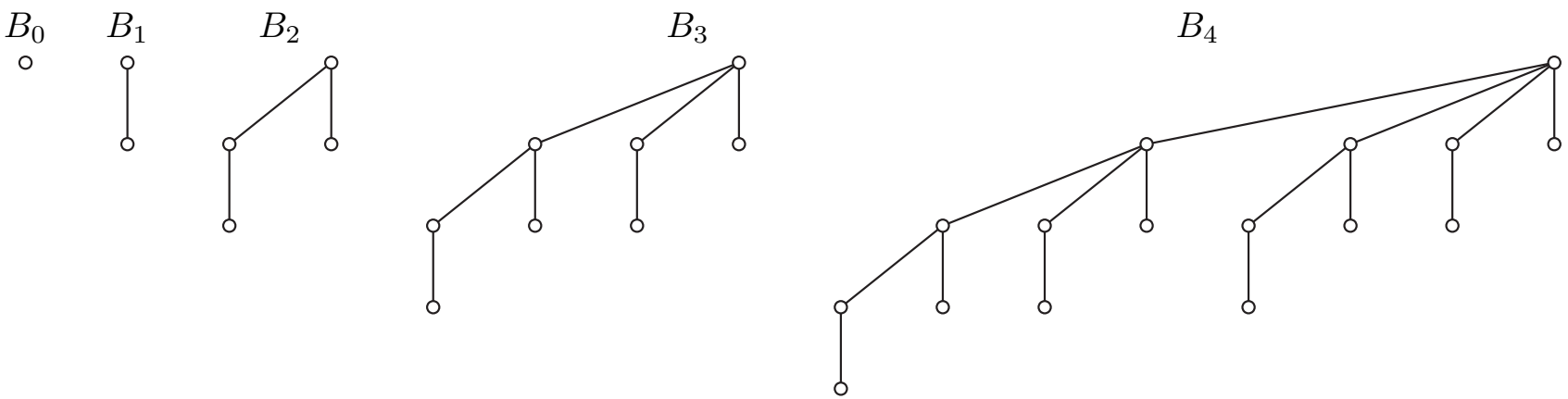

B_0
B_k
B_k
B_{k+1}



- každý takýto strom musí mať haldovité usporiadanie
- z každého stromu môžeme mať len jeden
- zjavne $|B_k| = 2^k$



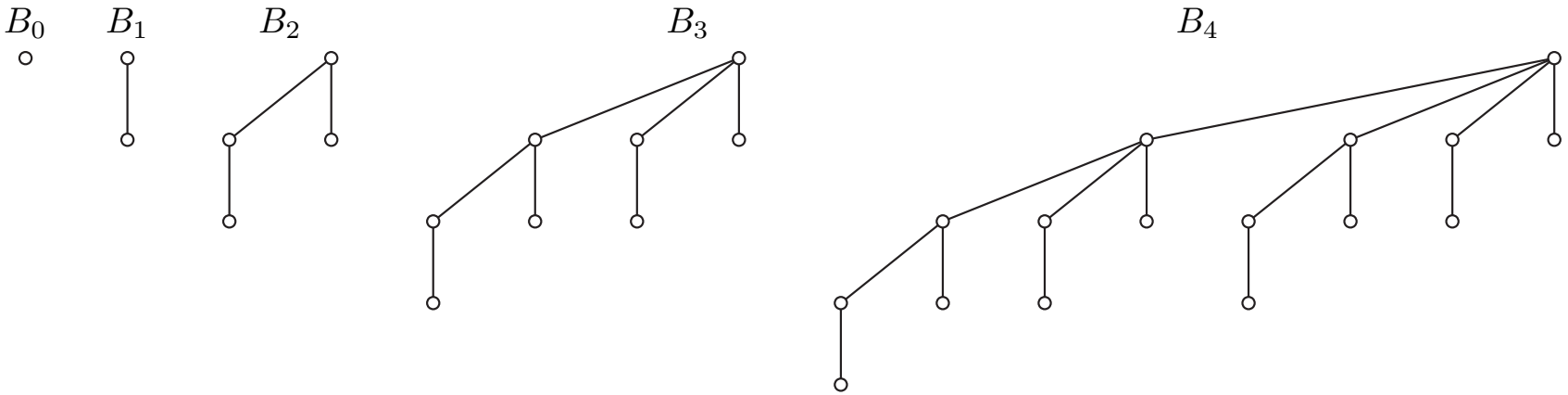
- každý takýto strom musí mať haldovité usporiadanie
- z každého stromu môžeme mať len jeden
- zjavne $|B_k| = 2^k$



Tvar haldy závisí od binárneho zápisu n:
Napr. $n = 41 = (101001)_2 = 32 + 8 + 1 = 2^5 + 2^3 + 2^0$,
$n = 75 = (1001011)_2 = 64 + 8 + 2 + 1 = 2^6 + 2^3 + 2^1 + 2^0$.

Dôsledky:

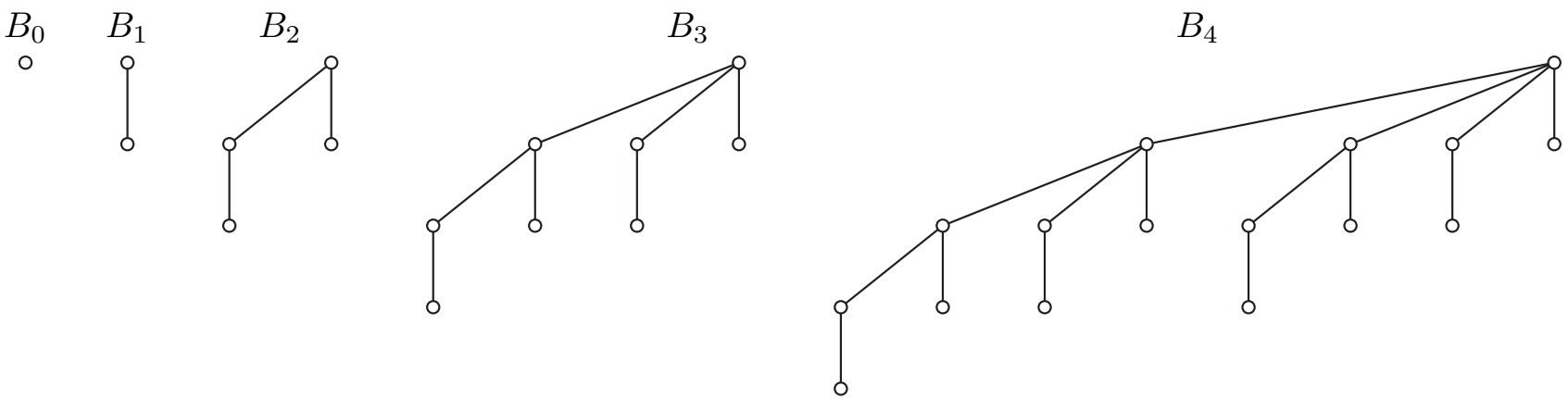
- najväčší strom B_k, ktorý halda obsahuje má $k \leq \log_2 n$
- stromov v halde je najviac $\log_2 n$



Tvar haldy závisí od binárneho zápisu n:
Napr. $n = 41 = (101001)_2 = 32 + 8 + 1 = 2^5 + 2^3 + 2^0$,
$n = 75 = (1001011)_2 = 64 + 8 + 2 + 1 = 2^6 + 2^3 + 2^1 + 2^0$.

Dôsledky:

- najväčší strom B_k, ktorý halda obsahuje má $k \leq \log_2 n$
- stromov v halde je najviac $\log_2 n$



Tvar haldy závisí od binárneho zápisu n:
Napr. $n = 41 = (101001)_2 = 32 + 8 + 1 = 2^5 + 2^3 + 2^0$,
$n = 75 = (1001011)_2 = 64 + 8 + 2 + 1 = 2^6 + 2^3 + 2^1 + 2^0$.

Dôsledky:

- najväčší strom B_k, ktorý halda obsahuje má $k \leq \log_2 n$
- stromov v halde je najviac $\log_2 n$

napr. 41+75=

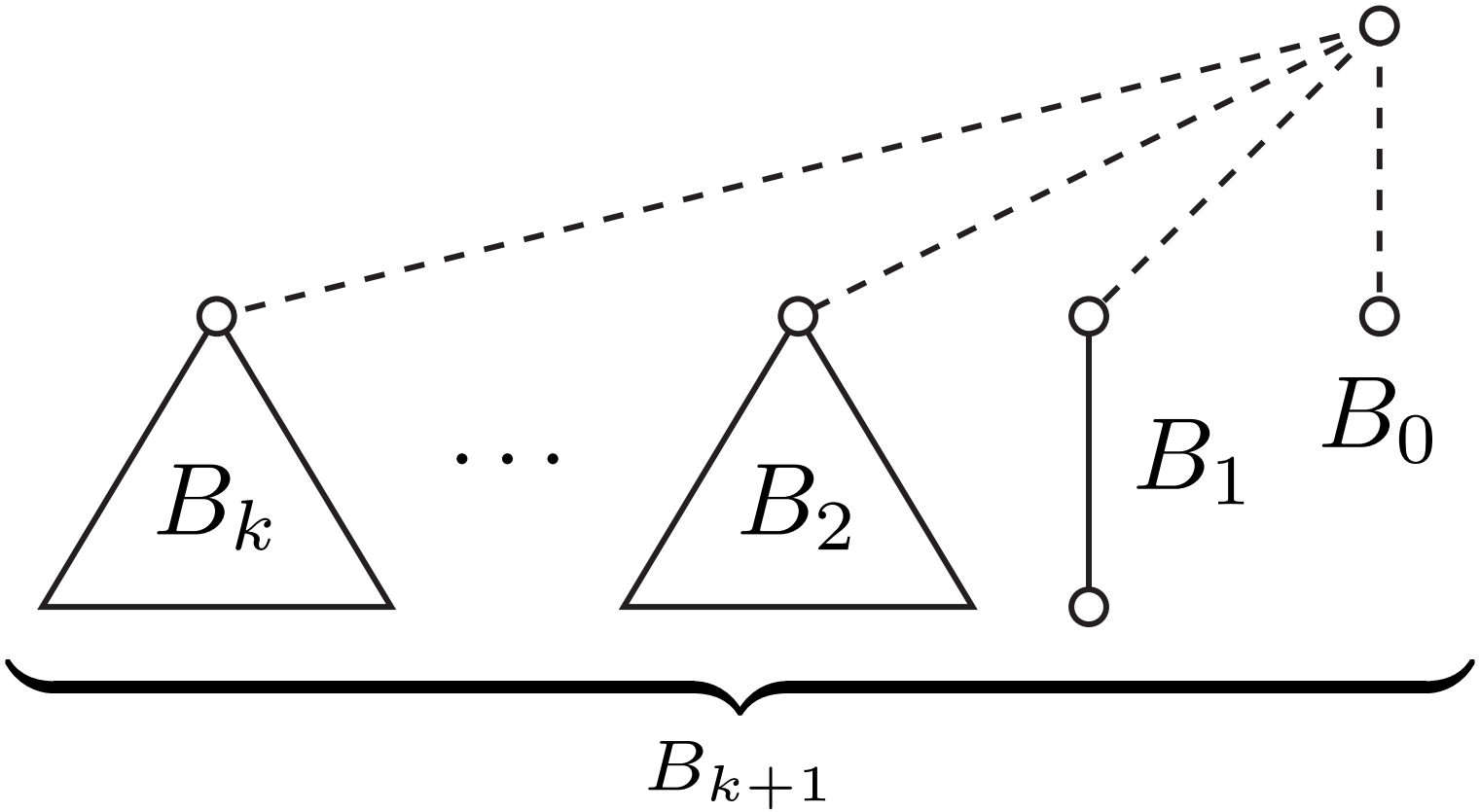
$$
\begin{array}{cccccccc}
 & & 1 & 0 & 1 & 0 & 0 & 1 \\
+ & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\
\hline
 & 1 & 1 & 1 & 0 & 1 & 0 & 0
\end{array}
$$

merge		B_5		B_3			B_0
with	B_6			B_3		B_1	B_0
	B_6	B_5	B_4		B_2		

napr. 41+75=

$$
\begin{array}{cccccccc}
 & & 1 & 0 & 1 & 0 & 0 & 1 \\
+ & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\
\hline
 & 1 & 1 & 1 & 0 & 1 & 0 & 0
\end{array}
$$

merge		B_5		B_3			B_0
with	B_6			B_3		B_1	B_0
	B_6	B_5	B_4		B_2		


B_k
$\cdots$
B_2
B_1
B_0
B_{k+1}

Lenivé haldy

kuko

7.10.2020

Vybrané partie z dátových štruktúr

	binom. halda
merge	$O(\log n)$
insert	$O(\log n)$
extract-min	$O(\log n)$
decrease-key	$O(\log n)$

	binom. halda	
merge	$O(\log n)$	$O(1)$ am.
insert	$O(\log n)$	$O(1)$ am.
extract-min	$O(\log n)$	$O(\log n)$ am.
decrease-key	$O(\log n)$	$O(\log n)$

	binom. halda	lenivá bin. halda
merge	$O(\log n)$	$O(1)$ am.
insert	$O(\log n)$	$O(1)$ am.
extract-min	$O(\log n)$	$O(\log n)$ am.
decrease-key	$O(\log n)$	$O(\log n)$

	binom. halda	lenivá bin. halda	
merge	$O(\log n)$	$O(1)$ am.	$O(1)$ am.
insert	$O(\log n)$	$O(1)$ am.	$O(1)$ am.
extract-min	$O(\log n)$	$O(\log n)$ am.	$O(\log n)$ am.
decrease-key	$O(\log n)$	$O(\log n)$	$O(1)$ am.

	binom. halda	lenivá bin. halda	Fibonacciho halda
merge	$O(\log n)$	$O(1)$ am.	$O(1)$ am.
insert	$O(\log n)$	$O(1)$ am.	$O(1)$ am.
extract-min	$O(\log n)$	$O(\log n)$ am.	$O(\log n)$ am.
decrease-key	$O(\log n)$	$O(\log n)$	$O(1)$ am.

Lenivá binomiálna halda

INVARIANT: Každý koreň stromu má 1$.

- `merge`: 1$
- `insert`: 1$
- `extract-min`: $2 \log n + 1$$

INVARIANT: Každý koreň stromu má 1$.

- `merge`: 1$
- `insert`: 1$
- `extract-min`: $2 \log n + 1$$

analýza `extract-min`:

- zmažeme min a pripojíme deti (1\$)
- každé dieťa sa stane nový koreň $\Rightarrow$ dostane 1\$ (spolu $\leq \log n$)
- upratanie t stromov trvá $O(t)$; rozdeľme stromy na dve skupiny:
  - $\ell = \#$stromov, ktoré prilinkujeme pod iné
  - $r = \#$koreňov, ktoré ostanú koreňmi aj po upratovaní
- korene, ktoré prilinkujeme si to sami zaplatia
- $r \leq \log n$ – toto zaplatíme z peňazí za operáciu

analýza `extract-min`:

- zmažeme min a pripojíme deti (1\$)
- každé dieťa sa stane nový koreň $\Rightarrow$ dostane 1\$ (spolu $\leq \log n$)
- upratanie t stromov trvá $O(t)$; rozdeľme stromy na dve skupiny:
  - $\ell = \#$stromov, ktoré prilinkujeme pod iné
  - $r = \#$koreňov, ktoré ostanú koreňmi aj po upratovaní
- korene, ktoré prilinkujeme si to sami zaplatia
- $r \leq \log n$ – toto zaplatíme z peňazí za operáciu

analýza `extract-min`:

- zmažeme min a pripojíme deti (1\$)
- každé dieťa sa stane nový koreň $\Rightarrow$ dostane 1\$ (spolu $\leq \log n$)
- upratanie t stromov trvá $O(t)$; rozdeľme stromy na dve skupiny:
  - $\ell = \#$stromov, ktoré prilinkujeme pod iné
  - $r = \#$koreňov, ktoré ostanú koreňmi aj po upratovaní
- korene, ktoré prilinkujeme si to sami zaplatia
- $r \leq \log n$ – toto zaplatíme z peňazí za operáciu

analýza `extract-min`:

- zmažeme min a pripojíme deti (1\$)
- každé dieťa sa stane nový koreň $\Rightarrow$ dostane 1\$ (spolu $\leq \log n$)
- upratanie t stromov trvá $O(t)$; rozdeľme stromy na dve skupiny:
  - $\ell = \#$stromov, ktoré prilinkujeme pod iné
  - $r = \#$koreňov, ktoré ostanú koreňmi aj po upratovaní
- korene, ktoré prilinkujeme si to sami zaplatia
- $r \leq \log n$ – toto zaplatíme z peňazí za operáciu

analýza `extract-min`:

- zmažeme min a pripojíme deti (1\$)
- každé dieťa sa stane nový koreň $\Rightarrow$ dostane 1\$ (spolu $\leq \log n$)
- upratanie t stromov trvá $O(t)$; rozdeľme stromy na dve skupiny:
  - $\ell = \#$stromov, ktoré prilinkujeme pod iné
  - $r = \#$koreňov, ktoré ostanú koreňmi aj po upratovaní
- korene, ktoré prilinkujeme si to sami zaplatia
- $r \leq \log n$ – toto zaplatíme z peňazí za operáciu

analýza `extract-min`:

- zmažeme min a pripojíme deti (1\$)
- každé dieťa sa stane nový koreň $\Rightarrow$ dostane 1\$ (spolu $\leq \log n$)
- upratanie t stromov trvá $O(t)$; rozdeľme stromy na dve skupiny:
  - $\ell = \#$stromov, ktoré prilinkujeme pod iné
  - $r = \#$koreňov, ktoré ostanú koreňmi aj po upratovaní
- korene, ktoré prilinkujeme si to sami zaplatia
- $r \leq \log n$ – toto zaplatíme z peňazí za operáciu

analýza `extract-min`:

- zmažeme min a pripojíme deti (1\$)
- každé dieťa sa stane nový koreň $\Rightarrow$ dostane 1\$ (spolu $\leq \log n$)
- upratanie t stromov trvá $O(t)$; rozdeľme stromy na dve skupiny:
  - $\ell = \#$stromov, ktoré prilinkujeme pod iné
  - $r = \#$koreňov, ktoré ostanú koreňmi aj po upratovaní
- korene, ktoré prilinkujeme si to sami zaplatia
- $r \leq \log n$ – toto zaplatíme z peňazí za operáciu

analýza `extract-min`:

- zmažeme min a pripojíme deti (1\$)
- každé dieťa sa stane nový koreň $\Rightarrow$ dostane 1\$ (spolu $\leq \log n$)
- upratanie t stromov trvá $O(t)$; rozdeľme stromy na dve skupiny:
  - $\ell = \#$stromov, ktoré prilinkujeme pod iné
  - $r = \#$koreňov, ktoré ostanú koreňmi aj po upratovaní
- korene, ktoré prilinkujeme si to sami zaplatia
- $r \leq \log n$ – toto zaplatíme z peňazí za operáciu

$$\Phi(H) = \#\text{bin. stromov} = \#\text{koreňov}$$

operácia `extract-min` pre haldu s t koreňmi trvá $O(t + \log n)$
$= O(\ell + \log n)$ zmena potenciálu $\Delta\Phi \leq \log n - \ell$

$$T_{amort} = T_{skut} + c \cdot \Delta\Phi = O(\ell + \log n) + c \cdot (\log n - \ell) = O(\log n)$$

$$\Phi(H) = \#\text{bin. stromov} = \#\text{koreňov}$$

operácia `extract-min` pre haldu s t koreňmi trvá $O(t + \log n)$
$= O(\ell + \log n)$ zmena potenciálu $\Delta\Phi \leq \log n - \ell$

$$T_{amort} = T_{skut} + c \cdot \Delta\Phi = O(\ell + \log n) + c \cdot (\log n - \ell) = O(\log n)$$

$$\Phi(H) = \#\text{bin. stromov} = \#\text{koreňov}$$

operácia `extract-min` pre haldu s t koreňmi trvá $O(t + \log n)$
$= O(\ell + \log n)$ zmena potenciálu $\Delta\Phi \leq \log n - \ell$

$$T_{amort} = T_{skut} + c \cdot \Delta\Phi = O(\ell + \log n) + c \cdot (\log n - \ell) = O(\log n)$$

$$\Phi(H) = \#\text{bin. stromov} = \#\text{koreňov}$$

operácia `extract-min` pre haldu s t koreňmi trvá $O(t + \log n)$
$= O(\ell + \log n)$ zmena potenciálu $\Delta\Phi \leq \log n - \ell$

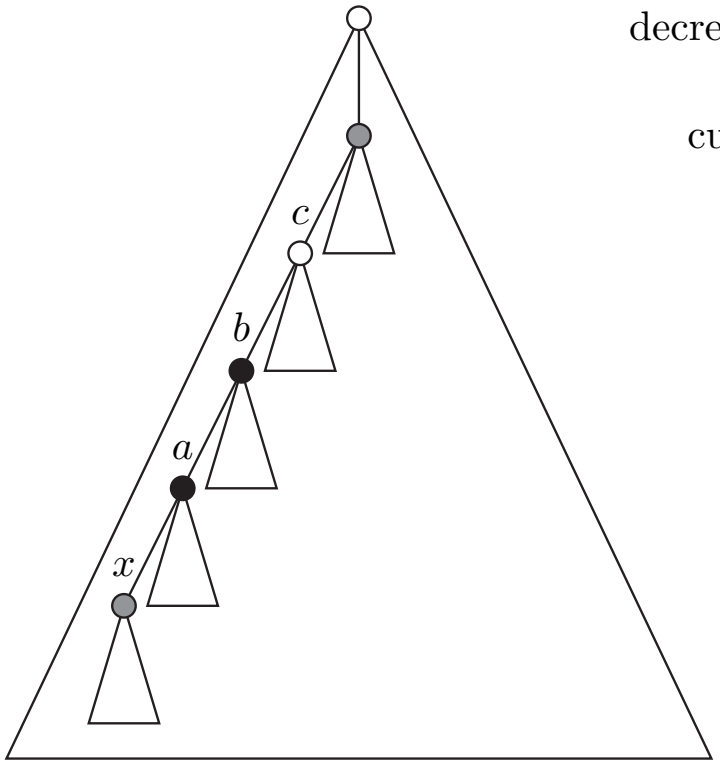
$$T_{amort} = T_{skut} + c \cdot \Delta\Phi = O(\ell + \log n) + c \cdot (\log n - \ell) = O(\log n)$$

$$\Phi(H) = \#\text{bin. stromov} = \#\text{koreňov}$$

operácia `extract-min` pre haldu s t koreňmi trvá $O(t + \log n)$
$= O(\ell + \log n)$ zmena potenciálu $\Delta\Phi \leq \log n - \ell$

$$T_{amort} = T_{skut} + c \cdot \Delta\Phi = O(\ell + \log n) + c \cdot (\log n - \ell) = O(\log n)$$

Fibonacciho halda



decrease key of x by δ

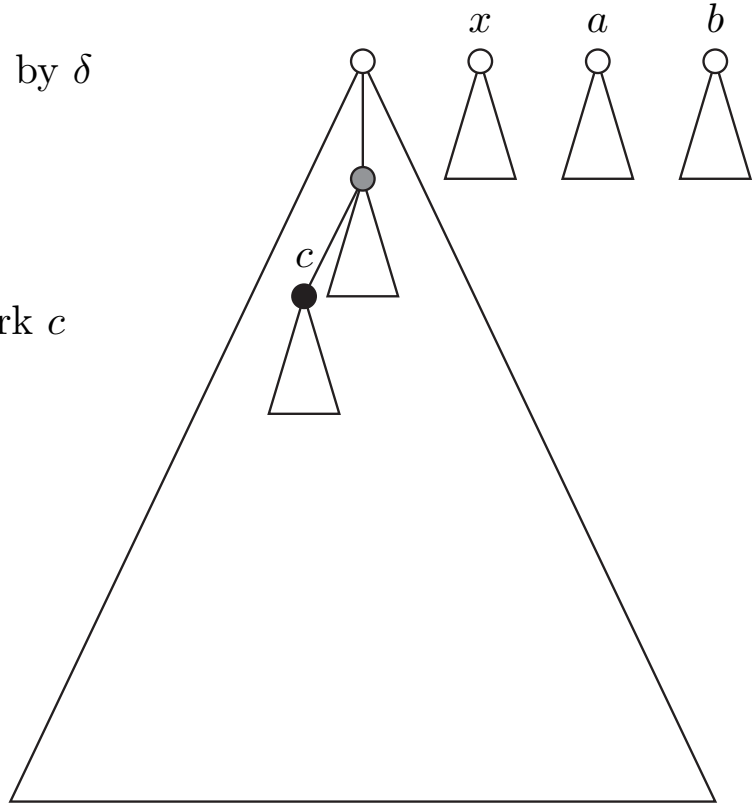
$\Longrightarrow$

cut x

$\rightarrow$ cut a

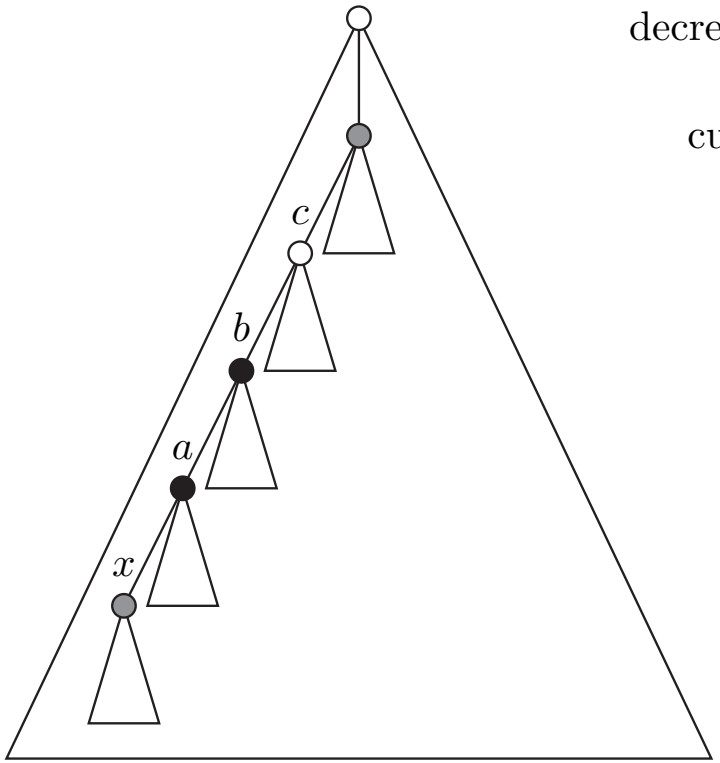
$\rightarrow$ cut b

$\rightarrow$ mark c



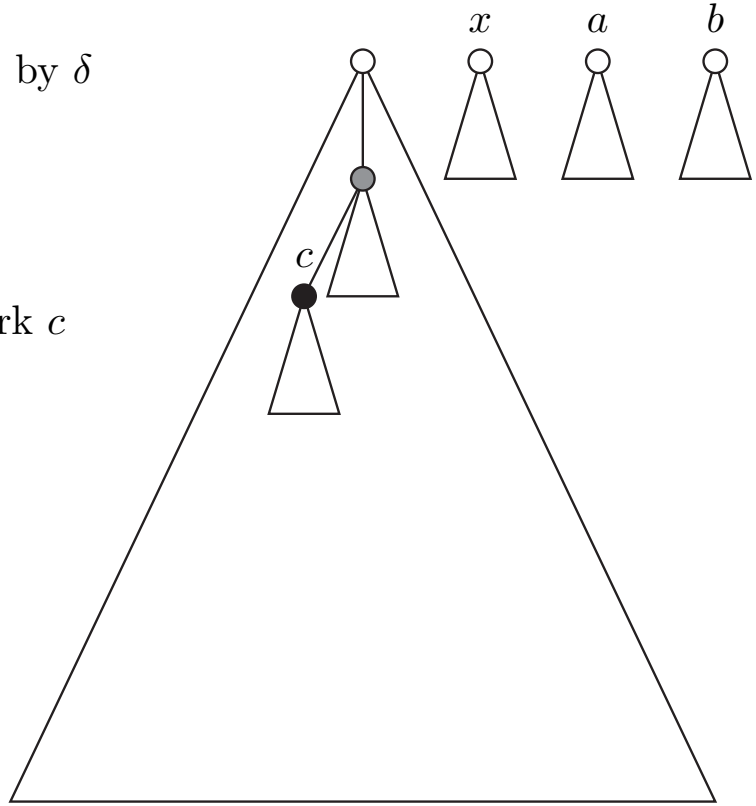
INVARIANT: každý označený vrchol, ktorý prišiel o syna, má nasporené 2$

na `decrease-key` stačia 4$:



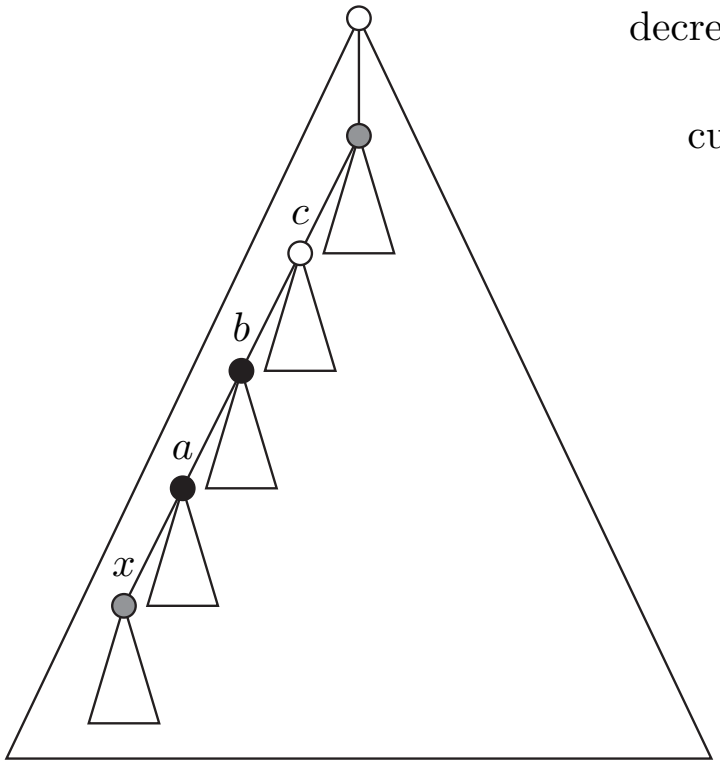
decrease key of x by δ
$\Longrightarrow$

cut x
$\rightarrow$ cut a
$\rightarrow$ cut b
$\rightarrow$ mark c

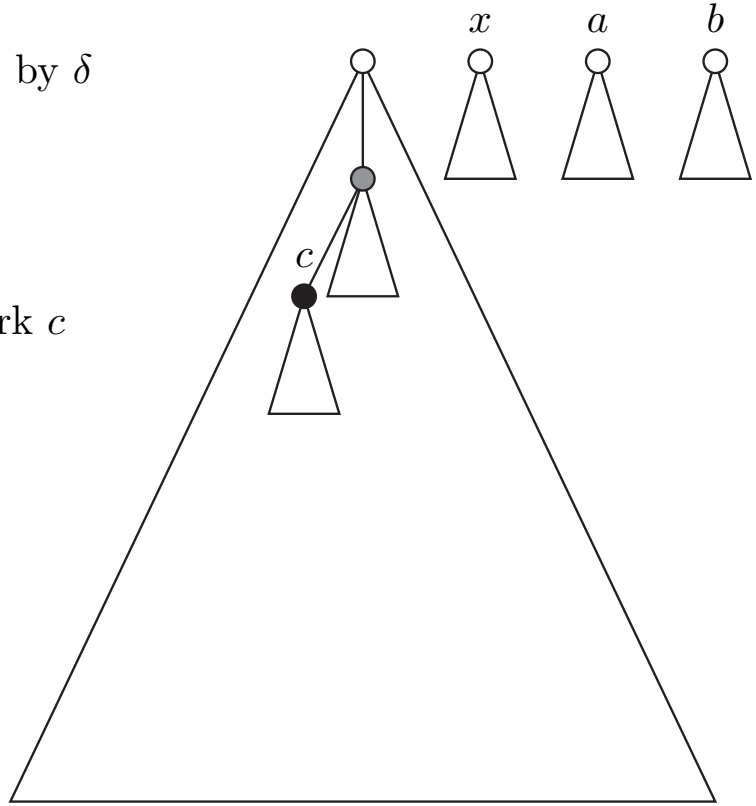


INVARIANT: každý označený vrchol, ktorý prišiel o syna, má nasporené 2$

na `decrease-key` stačia 4$:

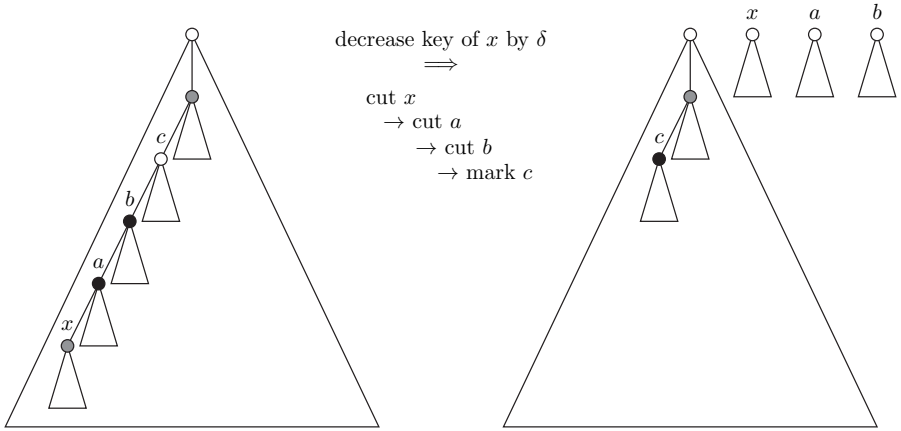


decrease key of x by δ
$\Longrightarrow$
cut x
$\rightarrow$ cut a
$\rightarrow$ cut b
$\rightarrow$ mark c



INVARIANT: každý označený vrchol, ktorý prišiel o syna, má nasporené 2$

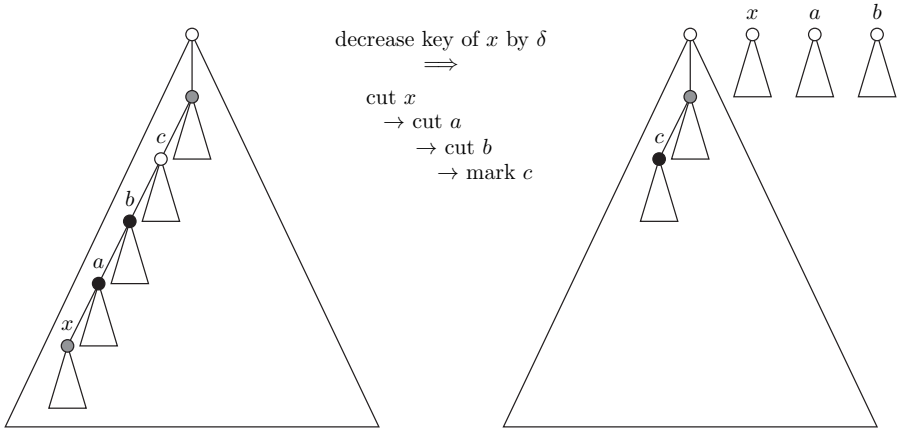
na `decrease-key` stačia 4$:



$$\Phi(H) = \#\text{koreňov} + 2 \times \#\text{označených vrcholov}$$

ak odrežeme k vrcholov, trvá to $O(k)$, ale potenciál klesne o $k-4$

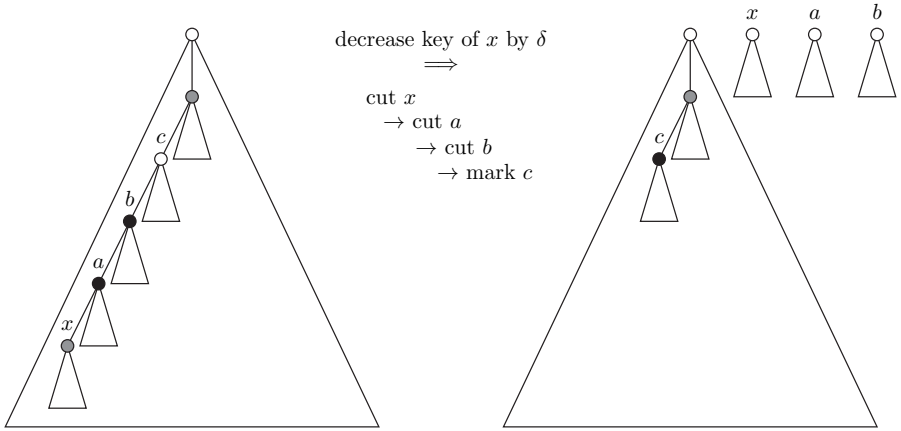
- $+k$ kvôli novým koreňom
- $-2 \times (k-1)$, lebo všetky okrem prvého koreňa boli označené a teraz nie sú
- $+2$ jeden označený na konci kaskády pribudol



$$\Phi(H) = \#\text{koreňov} + 2 \times \#\text{označených vrcholov}$$

ak odrežeme k vrcholov, trvá to $O(k)$, ale potenciál klesne o $k-4$

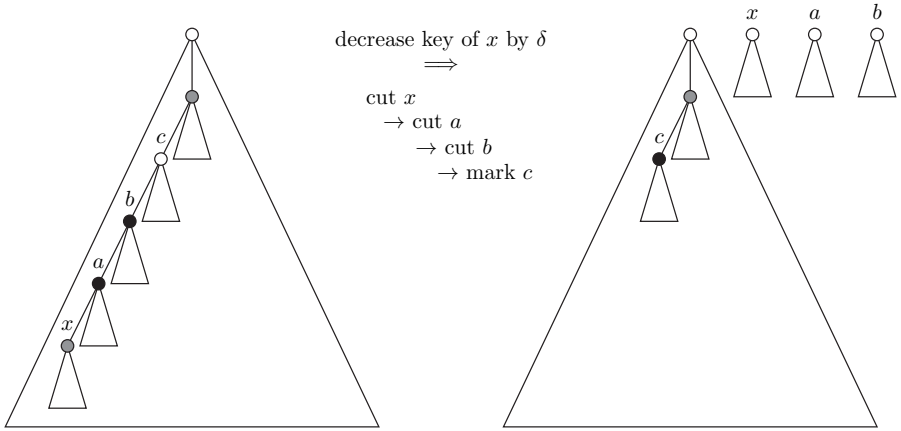
- $+k$ kvôli novým koreňom
- $-2 \times (k-1)$, lebo všetky okrem prvého koreňa boli označené a teraz nie sú
- $+2$ jeden označený na konci kaskády pribudol



$$\Phi(H) = \#\text{koreňov} + 2 \times \#\text{označených vrcholov}$$

ak odrežeme k vrcholov, trvá to $O(k)$, ale potenciál klesne o $k-4$

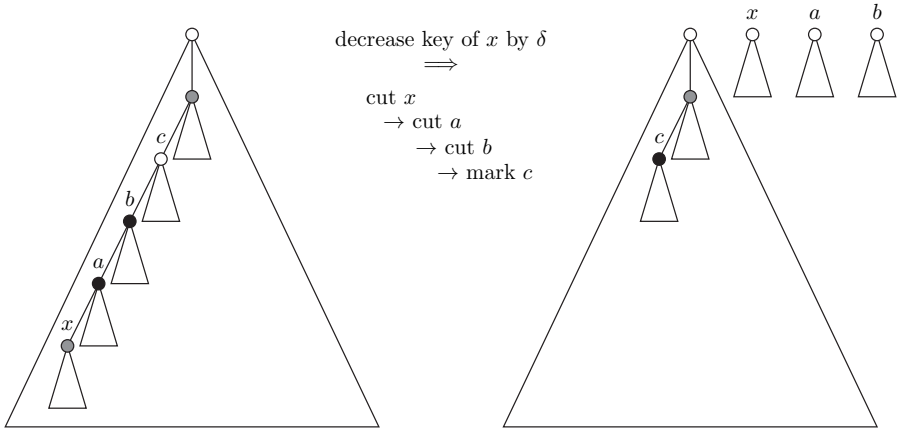
- $+k$ kvôli novým koreňom
- $-2 \times (k-1)$, lebo všetky okrem prvého koreňa boli označené a teraz nie sú
- $+2$ jeden označený na konci kaskády pribudol



$$\Phi(H) = \#\text{koreňov} + 2 \times \#\text{označených vrcholov}$$

ak odrežeme k vrcholov, trvá to $O(k)$, ale potenciál klesne o $k-4$

- $+k$ kvôli novým koreňom
- $-2 \times (k-1)$, lebo všetky okrem prvého koreňa boli označené a teraz nie sú
- $+2$ jeden označený na konci kaskády pribudol



$$\Phi(H) = \#\text{koreňov} + 2 \times \#\text{označených vrcholov}$$

ak odrežeme k vrcholov, trvá to $O(k)$, ale potenciál klesne o $k-4$

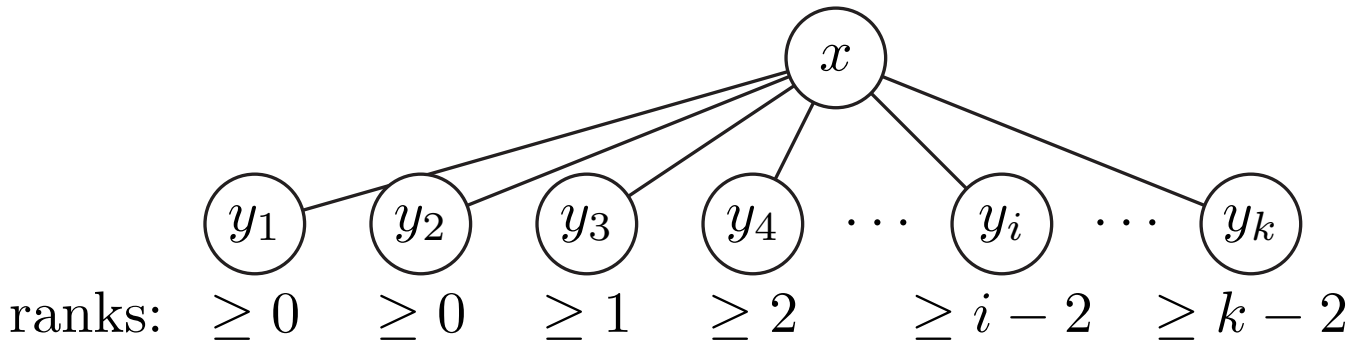
- $+k$ kvôli novým koreňom
- $-2 \times (k-1)$, lebo všetky okrem prvého koreňa boli označené a teraz nie sú
- $+2$ jeden označený na konci kaskády pribudol

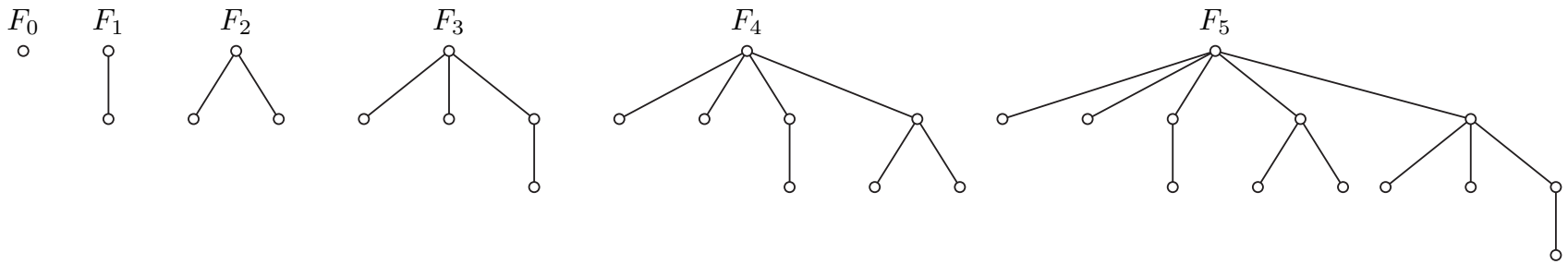
Veta

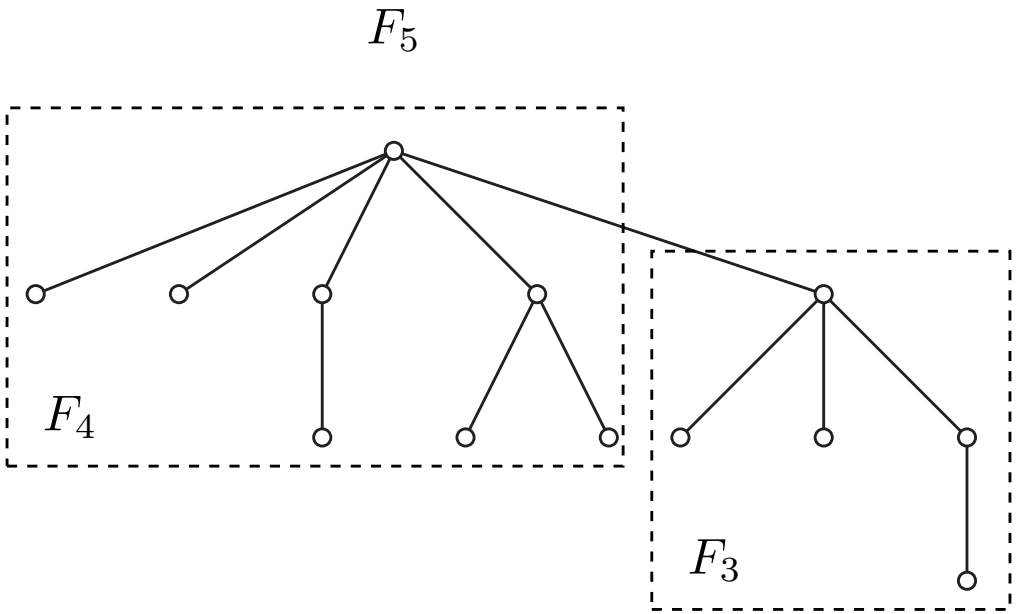
Nech $rank(v) = \#synov\ vrcholu.$
Pri takomto orezávaní bude mať vrchol s rankom k exponenciálne veľa vrcholov v závislosti od k.

Lema

Nech vrchol x má synov $y_1, \ldots, y_m$, ktorých sme označili v poradí, ako sa prilinkovali ku x; potom $\text{rank}(y_i) \geq i - 2$.



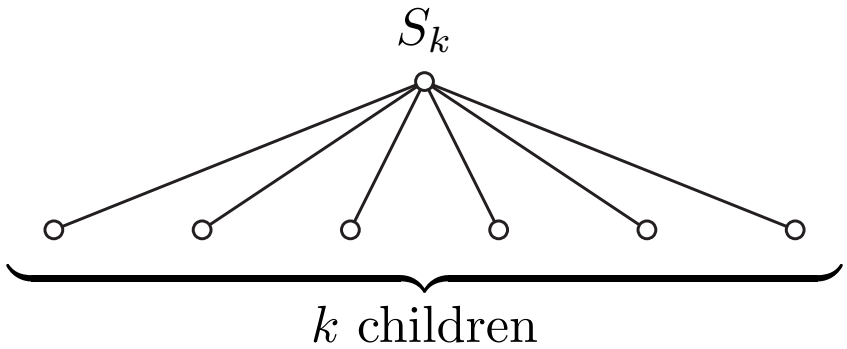

F_0
F_1
F_2
F_3
F_4
F_5

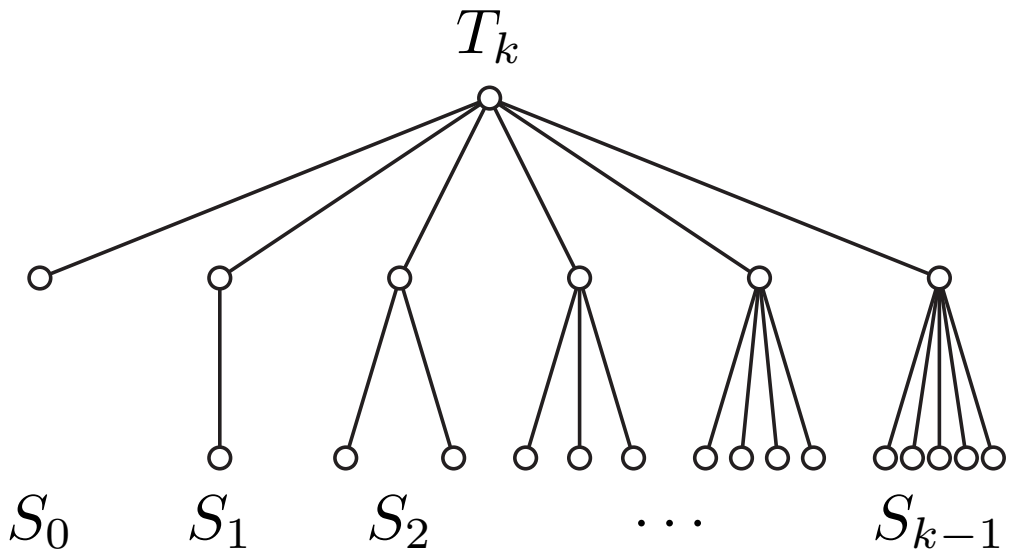

F_5
F_4
F_3

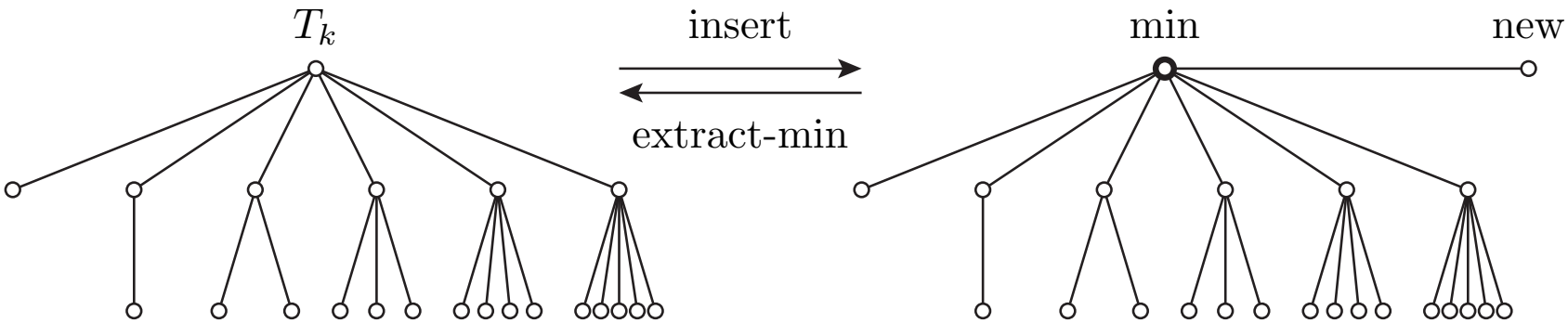
- veľkosť stromu s rankom k je aspoň $F_{k+1} \geq \phi^k$, kde $\phi \approx 1.618$
- $\Rightarrow$ maximálny rank je logaritmický
- $\Rightarrow$ každý koreň má najviac $O(\log n)$ synov a po uprataní ostane najviac $O(\log n)$ stromov

- veľkosť stromu s rankom k je aspoň $F_{k+1} \geq \phi^k$, kde $\phi \approx 1.618$
- $\Rightarrow$ maximálny rank je logaritmický
- $\Rightarrow$ každý koreň má najviac $O(\log n)$ synov a po uprataní ostane najviac $O(\log n)$ stromov

- veľkosť stromu s rankom k je aspoň $F_{k+1} \geq \phi^k$, kde $\phi \approx 1.618$
- $\Rightarrow$ maximálny rank je logaritmický
- $\Rightarrow$ každý koreň má najviac $O(\log n)$ synov a po uprataní ostane najviac $O(\log n)$ stromov


S_k
k children


T_k
S_0
S_1
S_2
$\cdots$
S_{k-1}


T_k
insert
extract-min
min
new