Pokażemy, że:

$$\begin{cases} \mu'(x) = \mu(x) \\ (**) \geqslant 0 \end{cases}$$ są sprzeczne

Z tego, że $\mu'(x) = \mu(x)$ oraz $(*)$ mamy $\mu(x) = \mu(y) = \mu(z) = \mu'(x)$

$(**) \geqslant 0 \Rightarrow \mu'(x) + \mu'(y) + \mu'(z) \geqslant \mu(x) + \mu(y) + \mu(z) = 3\mu'(x)$,

czyli $\mu'(y) + \mu'(z) \geqslant 2\mu'(x)$. Z tego oraz z $\mu'(z) \leqslant \mu'(y) \leqslant \mu'(x)$

mamy $\mu'(x) = \mu'(y) = \mu'(z)$

Tak więc z zał. mamy:

$$\mu(x) = \mu(y) = \mu(z) = \mu'(x) = \mu'(y) = \mu'(z)$$

Niech $a = |S(x)|$, $b = |$w nowym drzewie $z|$

$$a + b + 1 = |S(z)|$$

$$\lfloor \log(a) \rfloor = \lfloor \log(a+b+1) \rfloor = \lfloor \log(b) \rfloor$$

Niech $a \geqslant b$ wówczas

$$\lfloor \log(a+b+1) \rfloor > \lfloor \log(2b) \rfloor \geqslant \lfloor \log(b) \rfloor + 1$$

$\lightning$ sprzeczność

DRZEWCE (TREAPS)

9.05. 2018

$k_1 \leqslant k_2 \leqslant \ldots \leqslant k_n$

$p_1 \quad p_2 \quad \ldots \quad p_n$

- pary: klucz, priorytet w wierzchołkach
- drzewo binarne.
- względem klucza BST
- względem priorytetów kopiec

Fakt: Dla każdego zestawu kluczy i priorytetów istnieje drzewiec, który je przechowuje. Jeśli priorytety są unikalne, to jest tylko jeden taki drzewiec.

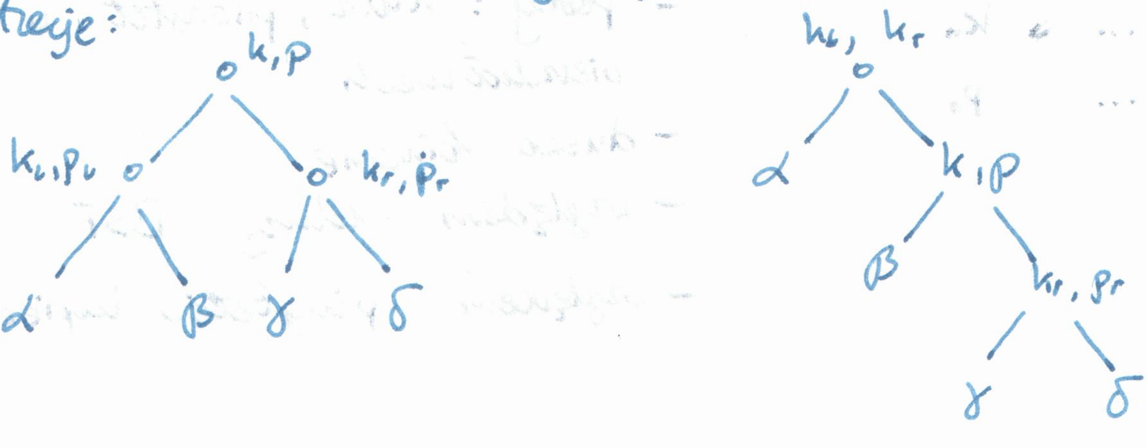
Drzewce losowe:

- p_i losujemy z rozkładem jednostajnym $[0,1)$

Operacje:

- wyszukiwanie : jak w BST
- wstawienie : wpierw jak w BST wstawiamy klucz, losujemy i rotacjami przywracamy porządek kopcowy
- usuwanie : wyszukiwanie jak w BST, rotacjami przesuwamy go do liścia i odcinamy (rotacja z synem, który ma większy priorytet)

Rotacje:



Analiza delete (m):

- klucze utożsamiamy ze zbiorem $\{1, 2, \ldots, n\}$ ← m

A - ścieżka prowadząca do m

koszt(del(m)) = |A| + |liczba rot|

Zajmiemy się oszacowaniem $E(|A|)$.

Ozn. $m_{\le}$ - zbiór kluczy $\le m$, analogicznie $m_{\ge}$

$|A| = |A \cap m_{\le}| + |A \cap m_{\ge}| - 2$ ← bo wierzch. m

Chcemy $E[|A|] = E[|A \cap m_{\le}|] + E[|A \cap m_{\ge}|] - 2$.

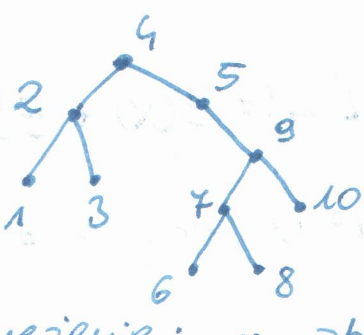
Oszacujemy $E[|A \cap m_{\le}|]$, drugie symetrycznie.

Ponieważ priorytety są losowe, to:

- permutacja σ kluczy powstała po uporządkowaniu ich malejąco wg priorytetów jest losowa

Przykład:

$\sigma = \langle 4, 5, 9, 2, 1, 7, 3, 10, 8, 6 \rangle$, niech $m = 8$



$A \cap m_{\le} = \{4, 5, 7, 8\}$

Spostrzeżenie: w zbiorze $A \cap m_{\le}$ znajdują się te klucze z $m_{\le}$, które:

- są w σ przed m
- są maksimami prefiksowymi z listy $m_{\le}$

Niech H_m - #maksimów prefiksowych ze zbioru $m_{<}$ w σ

$$X_1 = \begin{cases} 1, & \text{jeśli klucz 1 będzie maks. prefiksowe} \\ 0, & \text{w p.p.} \end{cases}$$

Y = # maksimów pref. ze zbioru $\{2, \dots, n\}$ w σ

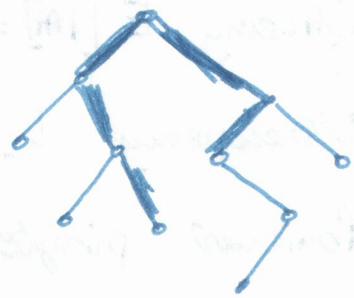
$$H_m = X_1 + Y$$

Wartość Y nie zależy od pozycji klucza 1 w σ, a więc $Y = H_{m-1}$

$$E[H_m] = E[X_1] + E[H_{m-1}] = \frac{1}{m} + E[H_{m-1}] = \sum \frac{1}{i} = O(\log m)$$

Oszacowanie # rotacji:

G_m = |długość skrajnie prawej ścieżki w ~~prawym~~ lewym poddrzewie| +
+ |długość skrajnie lewej ścieżki w prawym poddrzewie|



Oszacujemy G_m:

W tej ścieżce są klucze z $m_{<}$, które:

- są maks. pref. w $\sigma \cap \{1, \dots, m_{<}-1\}$
- w σ są na prawo od m

W podobny sposób jak poprzednio można pokazać, że $E[G_m] =$
$= O(\log n)$

Ćwiczenie: $E[G_m] < \cancel{2}1$

Analogicznie jak przy H_m mamy $E[G_m] = E(X) + E[G_{m-1}]$

Spostrzeżenie: klucz 1 jest ok $(X_1 = 1)$, gdy $\sigma = m, 1, \dots$

$E(X_1) = \frac{1}{m(m-1)}$. Można pokazać, że $E(G_m) = \frac{m-1}{m}$

UNION FIND

10.05.2018r.

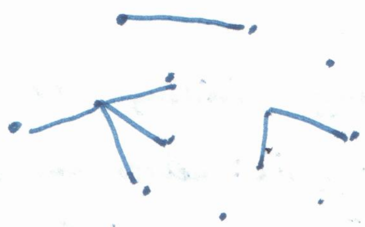
- universum $\mathcal{U} = \{1, \dots, n\}$
- rozłączne podzbiory rozbijające $\mathcal{U}$
- operacje:
  - FIND(x), $x \in \mathcal{U}$ (wynik podzbiór, do którego należy)
  - $\mathcal{U}$NION(_,_) - połączenie dwóch podzbiorów

Motywacje:

- MST (metoda Kruskala)

  n - wierzchołków m - krawędzi

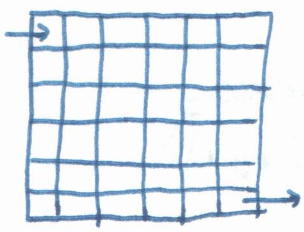
  m - # operacji FIND
  n - # operacji UNION



$e_1 \leq \dots \leq e_m$

$e_k = (v,u)$

- Konstrukcja labiryntu



na początku każde pole jest w różnej składowej, permutujemy kolejność i bierzemy po kolei pola, jeśli są w różnych składowych, to wymazujemy ścianę, jeśli nie, to zostawiamy

Rozw. naiwne:

R[0 :: |U|-1]

R[i] = "nazwa" podzbioru, do którego należy

FIND: return R[i] $O(1)$

UNION $O(n)$

czas wykonania σ $O(n+m^2)$

Ozn: σ - ciąg m operacji FIND i n operacji UNION

2 tryby wykonania σ:
- online
- offline (tzn. możemy się najpierw przyjrzeć całej σ)

Nas dzisiaj interesuje tryb online :)