Drzewa Splay

wtorek, 14 maja 2024 14:16

⇒ Struktura samoorganizująca się

⇒ drzewa BST

⇒ operacje:

1) insert
2) delete
3) find
4) join
5) split

4) join(S_1, S_2) - połącz S_1 i S_2 w jedno drzewo

drzewa

$\forall_{k \in S_1} \forall_{l \in S_2} \; k < l$

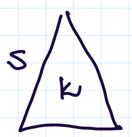
5) split(i, S) - rozdziel S na dwa drzewa S_1 i S_2 t.że

$\forall_{k \in S_1} \; k < i \quad \forall_{i \in S_2} \; k \geq i$

Operacja wewnętrzna SPLAY(S, k)

Przeorganizuj S tak, by k znalazł się w korzeniu (o ile $k \in S$) lub w korzeniu znalazł się j, t.że

$\forall_{v \in S} \; \neg\, j < v < k \;\wedge\; \neg\, k < v < j$



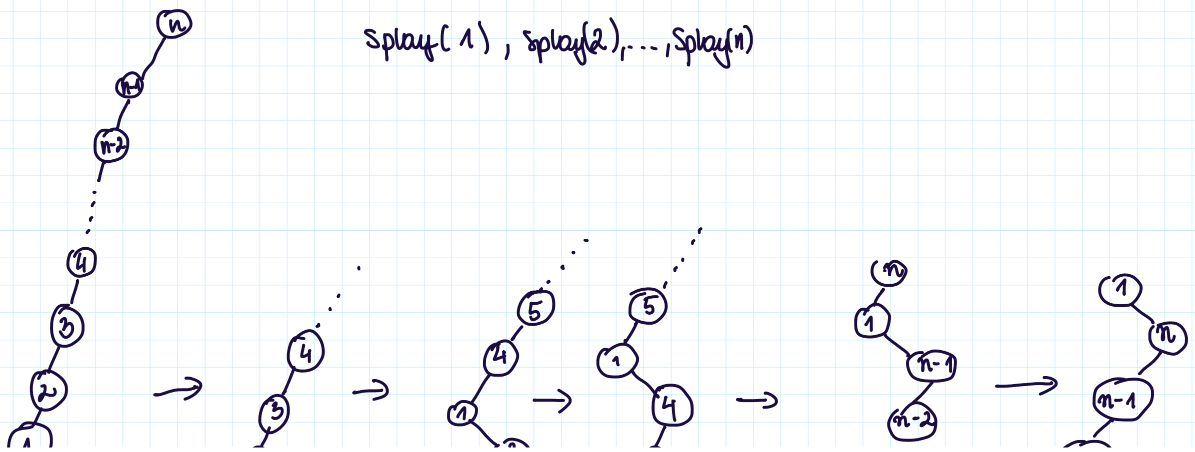
po Splay(S, k)

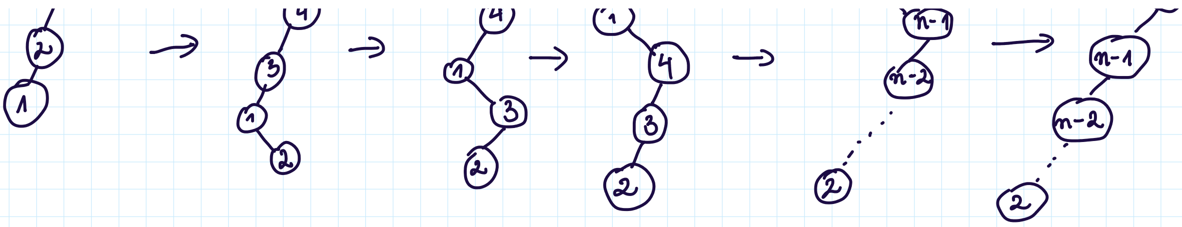
Naiwnie ⇒ rotacjami windujemy k do korzenia

Rozważmy to windowanie na przykładzie:

Splay(1)

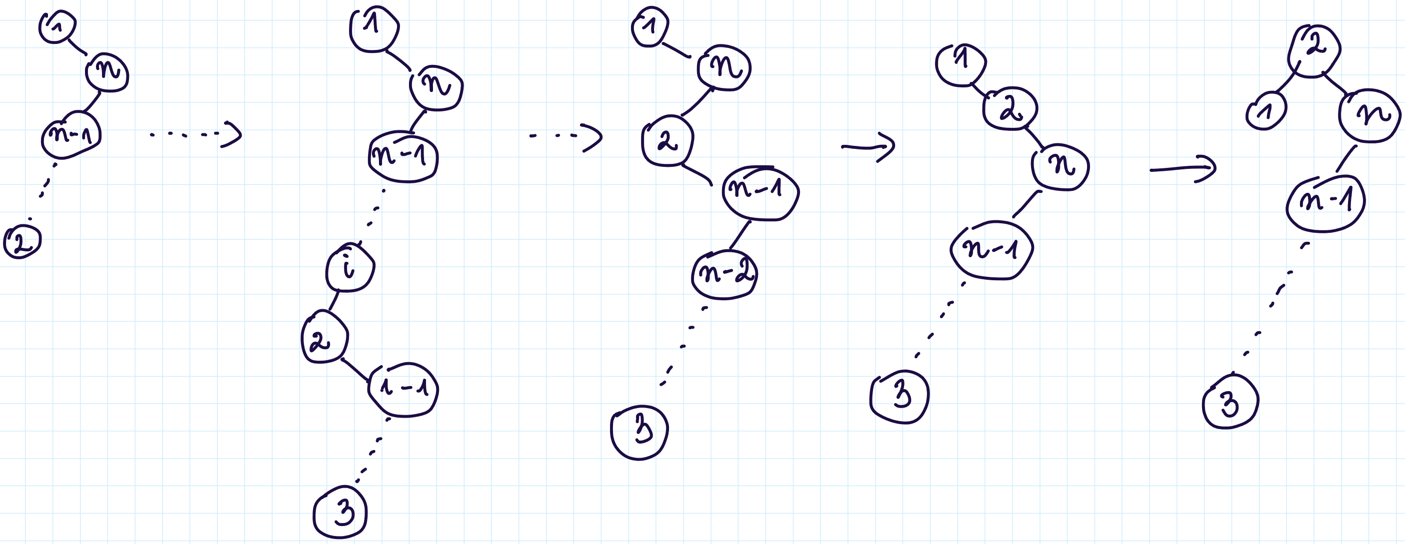
splay(1), splay(2), ..., Splay(n)





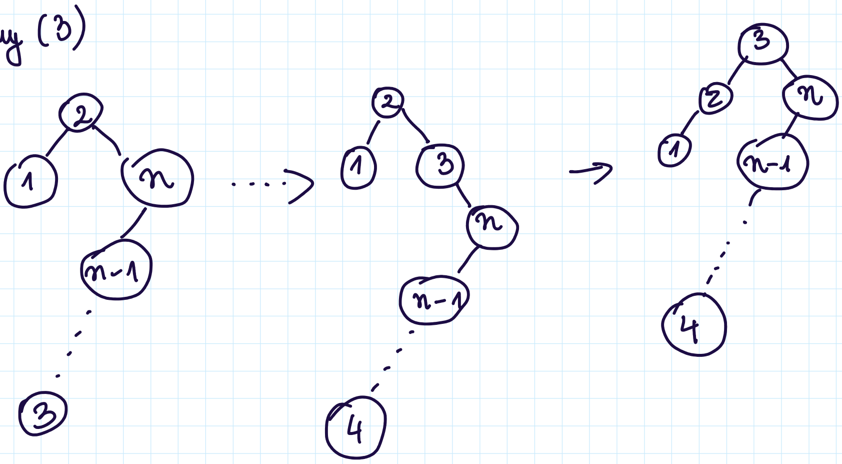
n-1 rotacji

SPLAY(2) :

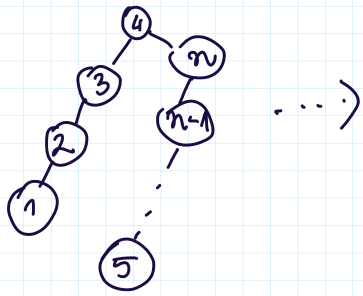


znowu n-1 rotacji

Splay (3)



Splay (4)



Splay(n)

σ Splay(1), Splay(2), ..., Splay(n)

Koszt σ wynosi $\Theta(n^2)$

Jak lepiej wykonywać Splay(x)?

Rozróżnimy trzy przypadki:

1) x nie ma dziadka (ale ma ojca)

Oznaczenie y - ojciec(x)

2) x jest lewym synem y i y jest lewym synem swego ojca lub x jest prawym synem y i y jest prawym synem swego ojca

3) x jest lewym synem y a y prawym synem swego ojca lub x jest prawym synem y a y lewym synem swego ojca.

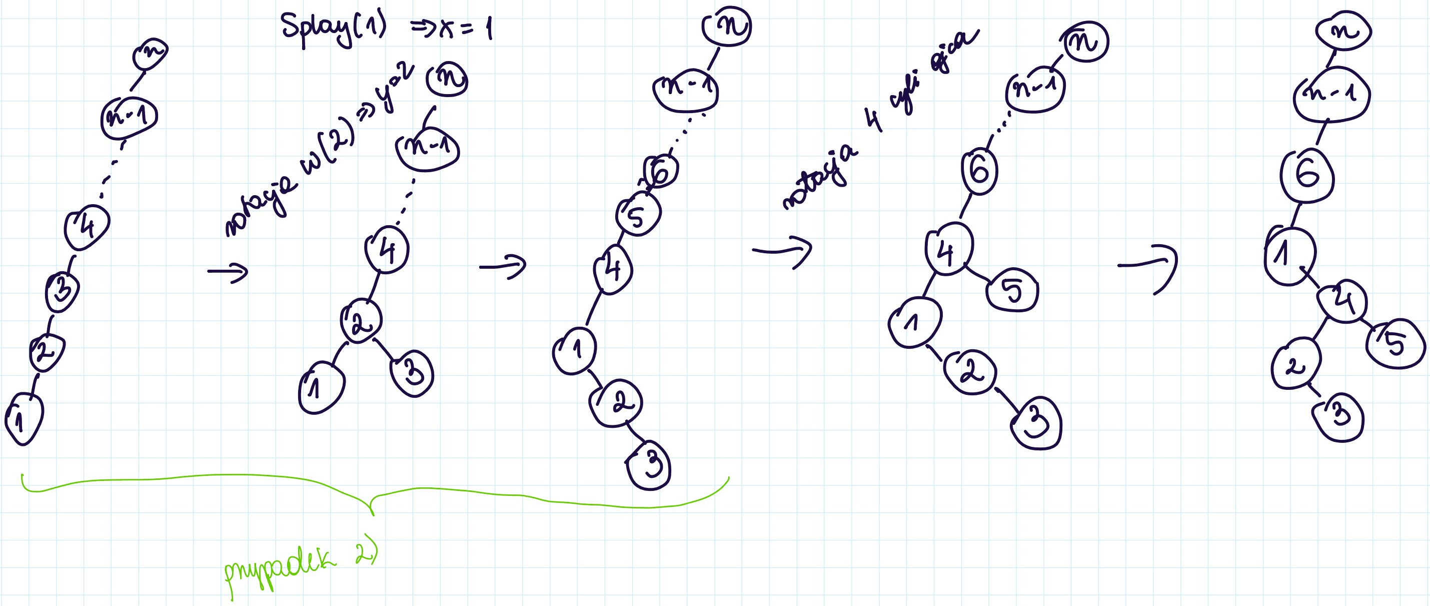
Jak postępujemy?

Ad 1) rotacja(x)

Ad 2) rotacja(y) a później rotacja(x)

Ad 3) dwa razy rotacja(x)

Splay(1) $\Rightarrow x = 1$



Ozn. x - wierzch. S(x) - poddrzewo o korzeniu x

$\mu(x) = \lfloor \log |S(x)| \rfloor$

↑

lg 2 wielkości drzewa wiszącego pod x

Będziemy analizować koszt ciągu operacji SPLAY.

Niezmiennik:

Każdy wierzchołek x ma co najmniej $\mu(x)$ jednostek na swoim koncie

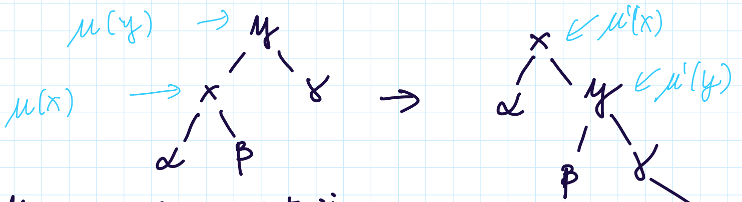
Spostrzeżenie:

każdą z operacji insert, find, delete, join, split można wykonać wykonując stałą liczbę operacji SPLAY i stałą liczbę operacji niskiego rzędu.

Lemat

Każda operacja SPLAY(x,S) wymaga nie więcej niż $3\cdot(\mu(S) - \mu(x)) + 1$ jednostek kredytowych do wykonania się i zachowania niezmiennika kredytowego

Przypadek 1: x nie ma dziadka



$\mu'(x)$ – μ dla x po wykonaniu rotacji

do utrzymania niezmiennika potrzebujemy:

Widać, że $\mu(y) = \mu'(x)$, zatem

$$(*) \quad \cancel{\mu'(x)} + \mu'(y) - \mu(x) - \cancel{\mu(y)}$$

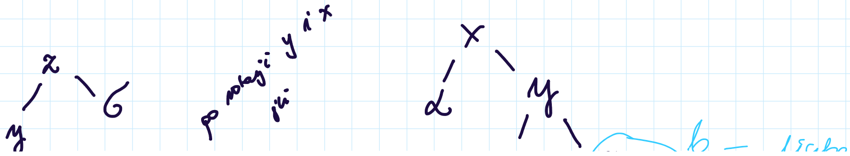
$$\leq \mu'(x) - \mu(x)$$

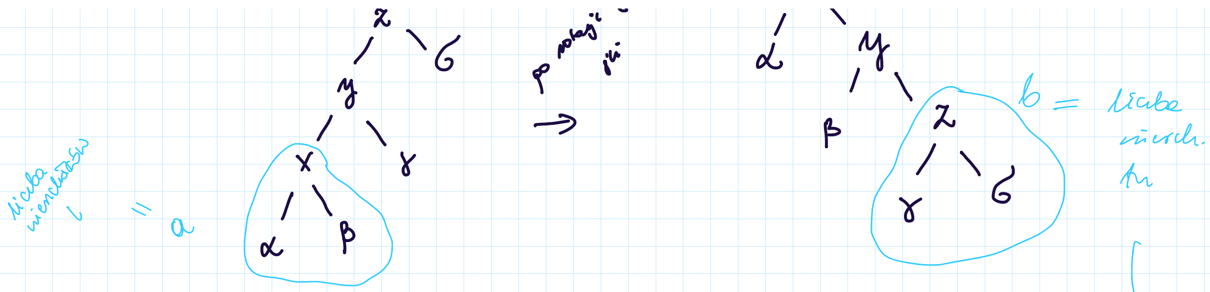
oraz $\mu'(y) \leq \mu'(x)$

$$\leq 3\cdot(\mu'(x) - \mu(x))$$

→ jak przypadek 1 to dodajemy jeszcze 1

Ad 2





Aby zachować niezmiennik, musimy zapłacić.

$$(*)\ \cancel{\mu'(x)} + \mu'(y) + \mu'(z) - \mu(x) - \mu(y) - \cancel{\mu(z)}$$

Obs: $\mu(z) = \mu'(x)$

$0 \le \mu(x) \le \mu(y) \le \mu(z)$

$0 \le \mu'(z) \le \mu'(y) \le \mu'(x)$

$$(*) = \mu'(y) + \mu'(z) - \mu(x) - \mu(y) \le 2\left(\mu'(x) - \mu(x)\right)$$

jeśli $\mu'(x) - \mu(x) > 0$ to mamy jedną różnicę $\mu'(x) - \mu(x)$ na opłacenie operacji niskiego rzędu

FAKT

jeśli $\mu'(x) = \mu(x)$ to $(*) < 0$

załóżmy nie wprost, że

$\mu'(x) = \mu(x)$ i

$\mu'(x) + \mu'(y) + \mu'(z) \geq \mu(x) + \mu(y) + \mu(z)$

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

z zał. $\mu'(x)$

stąd $\mu(x) = \mu(y) = \mu(z)$

możemy zastąpić

$\mu'(x) + \mu'(y) + \mu'(z) \geq 3 \cdot \mu(x)$

ale $\mu'(x) = \mu(x)$

$\mu'(y) + \mu'(z) \geq 2 \cdot \mu(x) = 2\mu'(x)$

stąd $\mu'(y) = \mu'(z) = \mu'(x)$

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

$$\mu(x) = \lfloor \lg a \rfloor$$

$$\mu'(x) = \lfloor \lg b \rfloor$$

$$\mu(z) = \lfloor \lg (a+b+1) \rfloor$$

$$\geqslant$$

$$\lfloor \lg (2 \cdot \min\{a,b\}+1) \rfloor$$

$$>$$

$$\lfloor \lg (\min\{a,b\}) \rfloor$$

SPRZECZNOŚĆ

$\square$

operacja insert wymaga wbicia początkowego depozytu do x.

$O(n \log n)$