Splay Trees

15-~~7~~50
~~1~~/23/16

Design-Goals A BST s.t.

1) No stored balance info.
2) The tree is "self-balancing".

Properties of a splay Tree.

1) Search(x) has side effect of rotating x to root.
   (Splay(x))
2) Subtle rotation rules.
3) Uses amortized analysis

Splay Trees

Two ways to describe them.

1) rotation rules

2) as Treap and rules to change priorities (insertion order)

eg

insertion order

(- - - b - - - - a - - - - -)   (- - - - ab - - - - - -)

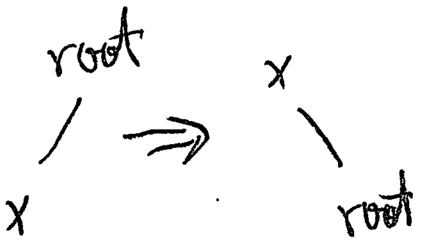
move a in front of b.

3 Rotation Rules for Splay

Goal Def Splay(x) ≡ rotate x to root.

Def Splay(x,T)

1) Zig ≡



(move x to front)

2) Zig-Zag

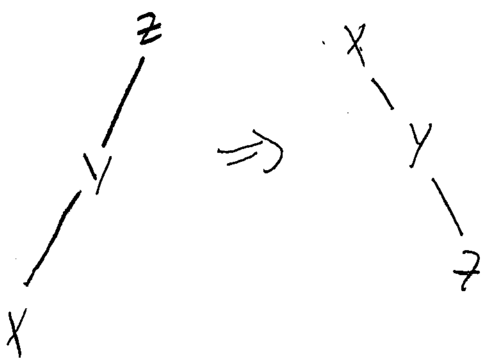
(move x in front of z)

or

1) (move w in front of z)
2) (move x " " w)

The interesting case

3) Zig-Zig



(move y in front of z
move x " " y)

Same as 2)

Priority version of splay:

Given x, y, z s.t. $\text{Parent}(x) = y$ $\text{Parent}(y) = z$

move y in front of z

move x " " y

Why not simple rule: move x in front of y.

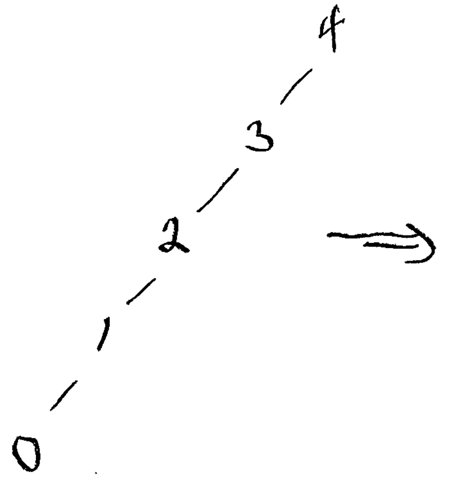
i.e. move x to front!

no zig-zig rule!

Keys 0, 1, 2, 3, 4

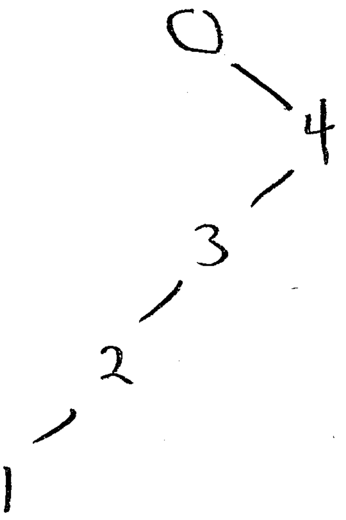
Priorities: (4, 3, 2, 1, 0)

Tree



move 0 to front ⇒

(0, 4, 3, 2, 1)

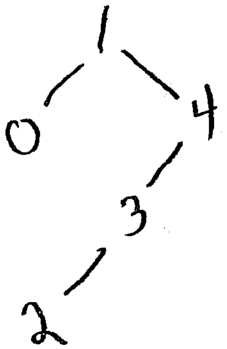


#rotations = 4

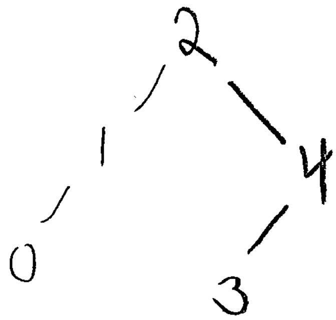
#rotations = 4

move 1 to front:

Priorities (1, 0, 4, 3, 2)

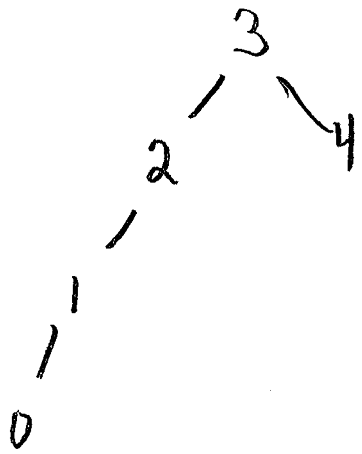


MTF 2

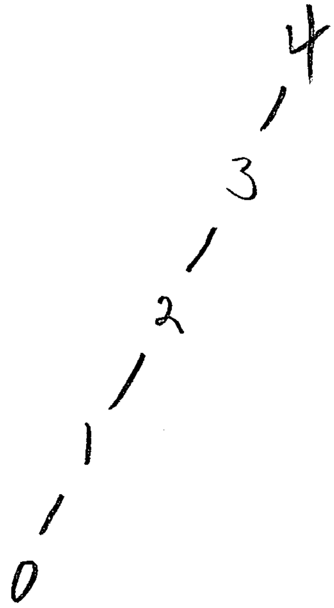


rotations = 3

rotations = 2



$\Rightarrow$



Priorities: $(n, n-1, \ldots, 0)$

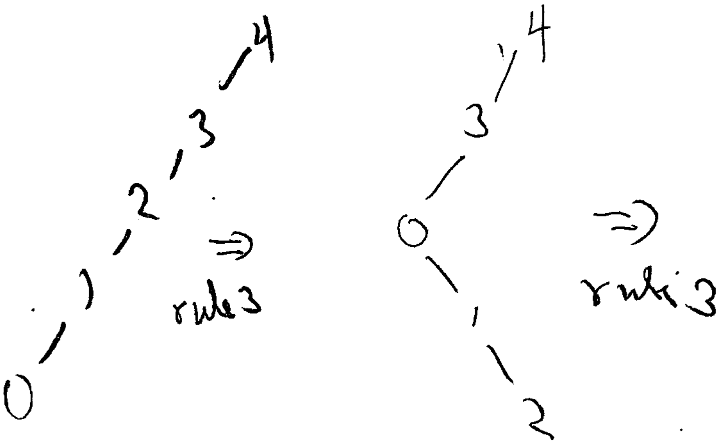
	MTF 0,			----	, MTF n	$n+1$ ops
rotations	n	n	$n-1$	...	1	$\Omega(n^2)$ rotations

Average # rotations $\Theta(n)$.

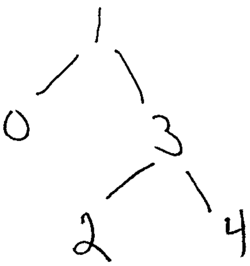
Using all 3 rules

input (4,3,2,1,0)

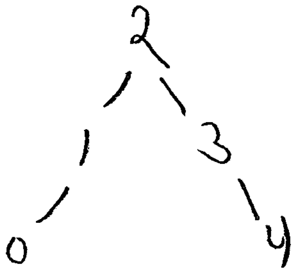
splay(0)



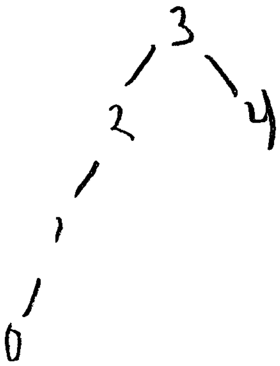
splay(1)



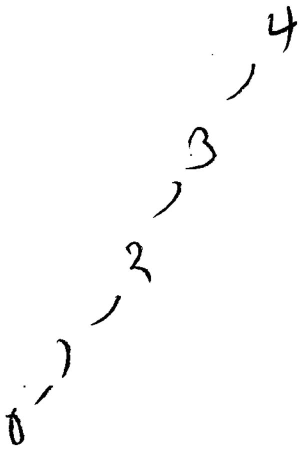
splay(2)



splay(3)



splay(4)

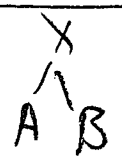


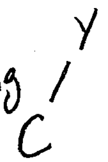
Dictionary Ops

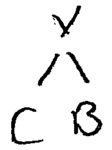
Claim

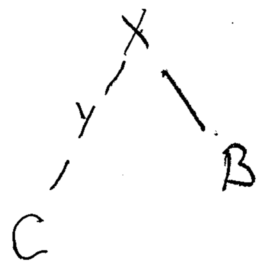
$AC = O(\lg n)$

Insert$(T, x) \equiv$ 1) VanillaInsert(T, x)
2) Splay(T, x)

Delete$(T, x) \equiv$ 1) Splay(T, x) giving 

2) if $A = \emptyset$ then return B
else splay(A, largest element) giving 

return 



Lookup$(T, x) \equiv$ 1) Search(T, x)
2) if x found then Splay(T, x)
else Splay$(T$, Parent$(x))$ (why?)

Amortized Analysis of Splay

Potential Method $\Phi(T)$

Goal: Φ large for unbalanced trees

Def $S(x)$ = # nodes in subtree rooted at x

Def $r(x) = \lfloor \log_2 (S(x)) \rfloor$ Rank

$$\Phi(T) = \sum_{x \in T} r(x)$$

$T \equiv$ Line then $\Phi(T) = \sum_{i=1}^{n} \lfloor \log i \rfloor = \Theta(n \log n)$

$T \equiv$ balanced then $\Phi(T) = \Theta(n)$

Claim $\forall T$ $\Phi(T) = \Omega(n)$ & $\Phi(T) = O(n \log n)$

$$\text{Amortized Cost} = \text{Unit-Cost} + \Delta\Phi$$

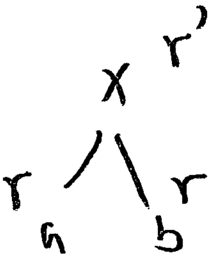
$$\text{Unit-Cost} \equiv \#\text{ rule applications} = \left\lceil \frac{\text{depth}}{2} \right\rceil \approx \frac{\#\text{rotations}}{2}$$

Access Lemma $AC(\text{splay}(x)) \le 3(r(\text{root}) - r(x)) + 1$

Cor $AC \le 3 \log n + 1$

Simple facts

Rank Rule



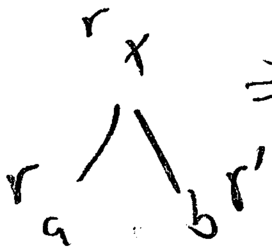
$\Rightarrow r' > r$

pf $s(a), s(b) \ge 2^r$

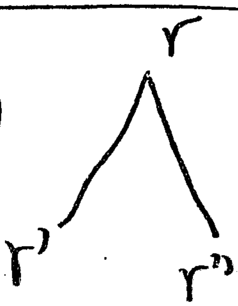
$\Rightarrow s(x) \ge 2^{r+1}$

$\Rightarrow r(x) \ge r+1$

ContraPositive



$\Rightarrow r' < r$ (or)



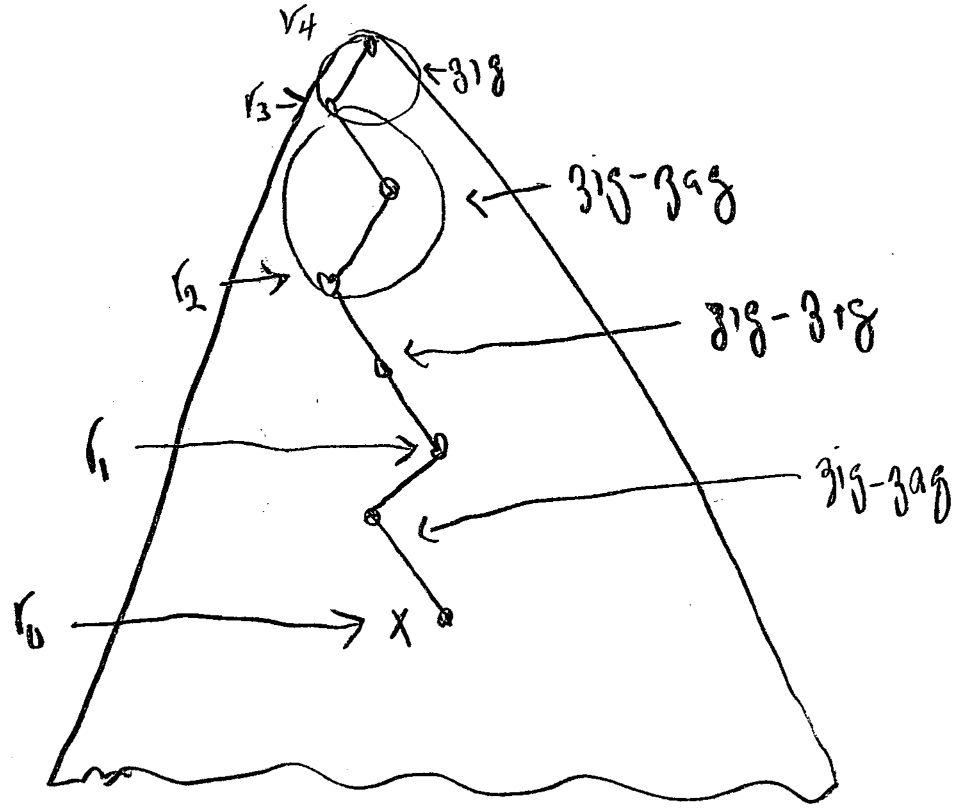
$2r > r' + r''$

Proof of Access Lemma

Lemma $AC(\text{zig}) \leq 3(r(\text{root}) - r(\text{child})) + 1$

Lemma $AC(\text{zig-zig}), AC(\text{zig-zag}) \leq 3(r(GP) - r(c))$

Proof by example splay(x)



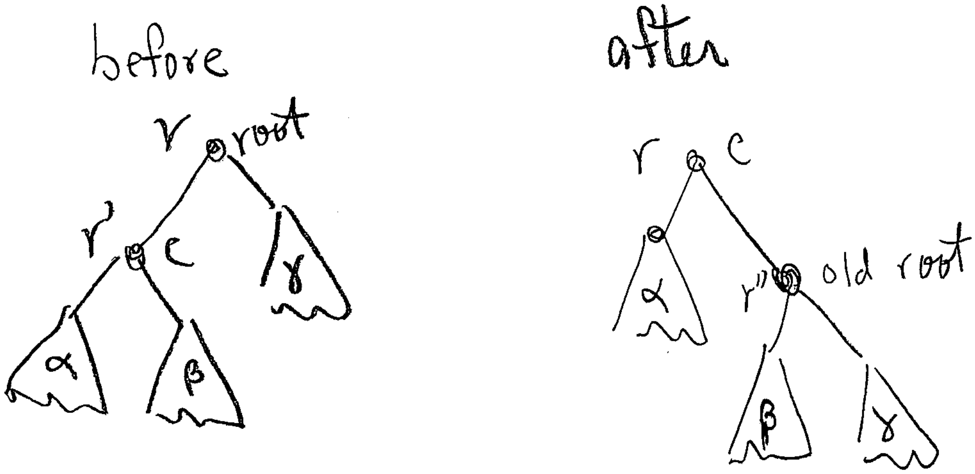
$$AC(\text{splay}) = AC(\text{zig-zag}) + AC(\text{zig-zig}) + A(\text{zig-zag}) + A(\text{zig})$$

$$\leq 3(r_1 - r_0) + 3(r_2 - r_1) + 3(r_3 - r_2) + 3(r_4 - r_3) + 1$$

$$\leq 3(r_4 - r_0) + 1$$

proof of zig-lemma

$$AC(zig) \le 3(r(root) - r(c)) + 1$$
$$= 3(r - r') + 1$$

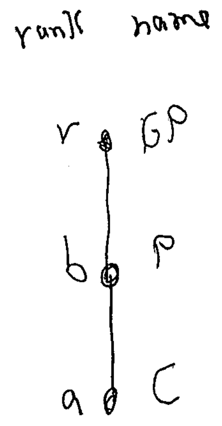


$$\Phi(after) = r + r'' + \Phi(\alpha) + \Phi(\beta) + \Phi(\gamma) = r + r'' + \Phi(subtrees)$$

$$\Phi(before) = r + r' + \Phi(subtrees)$$

$$AC(zig) = 1 + \Delta\Phi = 1 + r'' - r' \le 1 + (r - r') \le 3(r - r') + 1$$

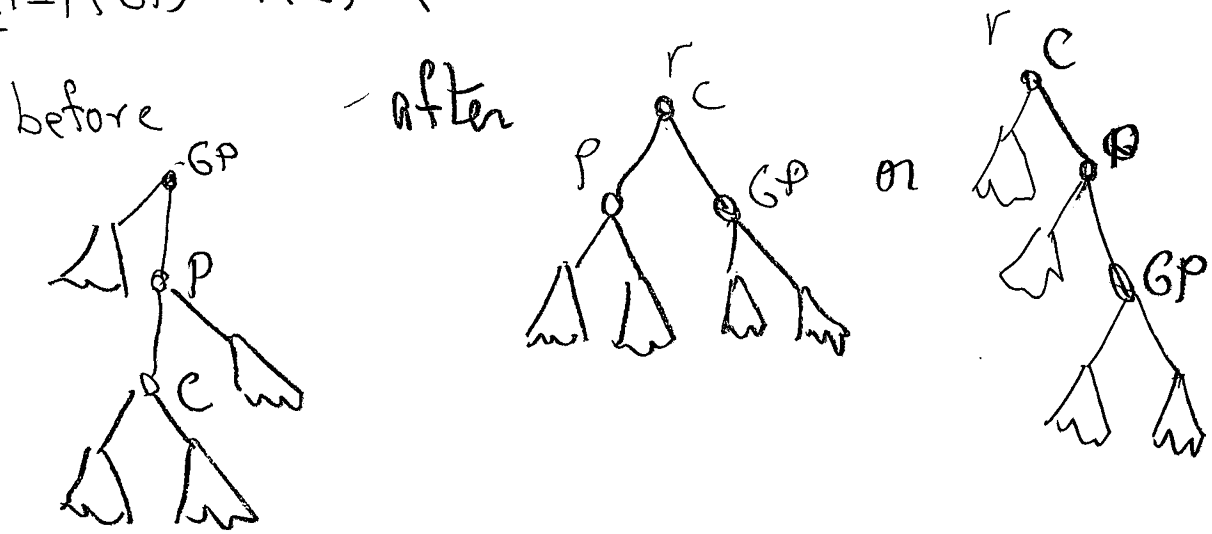
Table of potential change



Claim

		zig-zag : zig-zig
if	$r = r(GP) > r(C) = a$	$\Delta\Phi \le 2(r-a)$
if	$r = r(GP) = r(C)$	$\Delta\Phi \le -1$

Case $r = r(GP) > r(C) = a$



$$\Phi(\text{after}) \le 3r + \Phi(\text{subtrees})$$

$$\Phi(\text{before}) \ge r(GP) + r(P) + r(C) + \Phi(\text{subtrees})$$
$$\ge r + 2a + \Phi(\text{subtrees})$$

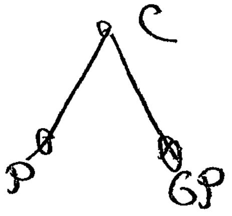
$$\Delta\Phi \le 3r - (r + 2a) = 2(r-a)$$

Case $r = r(GP) = r(C)$

$\Phi(\text{before}) = 3r + \Phi(\text{subtrees})$

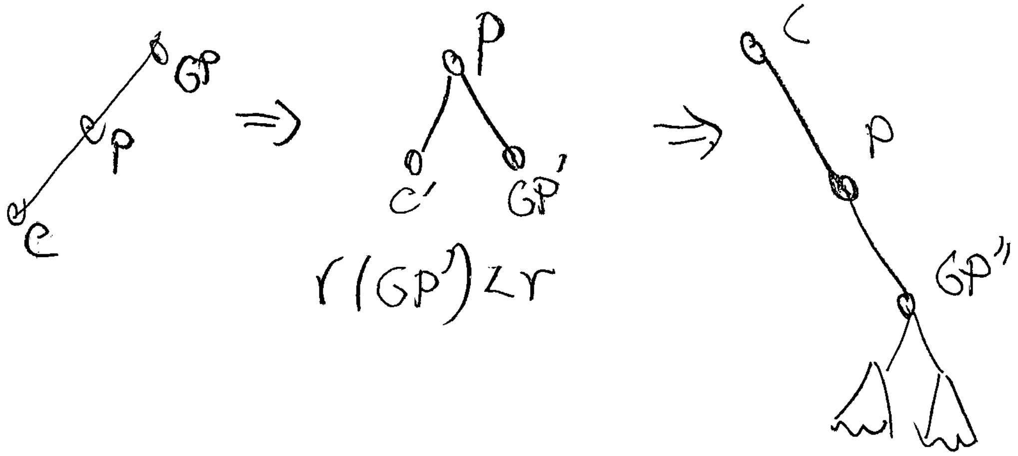
<u>Claim</u> $\Phi(\text{after}) \le 3r - 1 + \Phi(\text{subtrees})$

after
(3) 3-zag



$\Rightarrow r(P)$ or $r(GP) < r$

<u>after</u>



$r(GP') < r$

$\Delta\Phi = \Phi(\text{after}) - \Phi(\text{before})$

≤ -1

$$AC\left(\text{case } r(GP) > r(c)\right) = 1 + \Delta\underline{\Phi} \leq 1 + 2(r-a)$$
$$\leq 3(r-a)$$

$$AC\left(\text{case } r(GP) = r(c)\right) = 1 + \Delta\underline{\Phi} \leq 1 - 1 = 0 \leq 3(r-r)$$

Finished Access Lemma!

<u>Balance Thm</u> m splays on an n-node tree

splay does $O(m \log n + n \log n)$ rotation

<u>Pf</u> $AC \le 3 \log n + 1$ $\quad \Phi_{end} \ge 0$ $\quad \Phi_{begin} \le O(n \log n)$

$$\#\text{rotation} \le O\left(m(3 \log n + 1) + \Phi_{begin} - \Phi_{end}\right)$$

$$\le O(m \log n + n \log n)$$

Important extensions

Static Optimality Thm

m searches $\wedge$ q_i = # searches of K_i eg $m = \sum q_i$

then total cost $= O\left(m + \sum_{q_i > 0} q_i \log(m/q_i)\right)$

pf

Redo access lemma with weighted nodes

Set $S(x) = \sum_{y \in \text{Subtree}(x)} w(y)$

Set $\text{rank}(x) = \log(S(x))$

Claim $A((\text{splay}(x)) \leq 3(\text{rank}(\text{root}) - \text{rank}(x)) + 1$

For Opt proof set $w(x) = q_i/m$