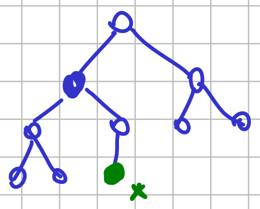
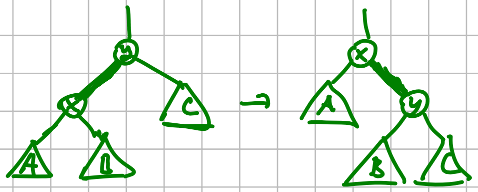
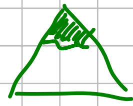
Splay tree



Rotation





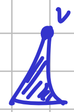
[Sleator & Tarjan 1983]

Goal: prove that complexity of Splay is $O(\log n)$ amortized.

Rule: When accessing x, bring it to the root by sequence of splay steps
- zig-zig
- zig-zag
- zig } only as last

} Splaying of the node

Notation:



$T(v)$ ... subtree rooted at v

$s(v)$ ... size of v $\quad s(v) := |T(v)|$

$r(v)$ ... rank of v $\quad r(v) := \log s(v)$ ← binary

Φ ... potential $\quad \Phi := \sum_v r(v)$

n ... # nodes in the whole tree

$1 \le s(v) \le n$

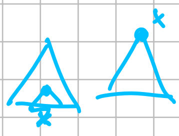
$0 \le r(v) \le \log n$

$0 \le \Phi \le n \log n$

Real cost of a Splay := # rotations (zig-zig / zig-zag has cost 2)

Theorem: Amortized cost of Splay(x) is at most $3(r'(x) - r(x)) + 1$.

$r'(x)$ … rank after, $r(x)$ … rank before



↳ it's $O(\log n)$.

Claim: if $r_i(x) := r(x)$ after i-th splay step

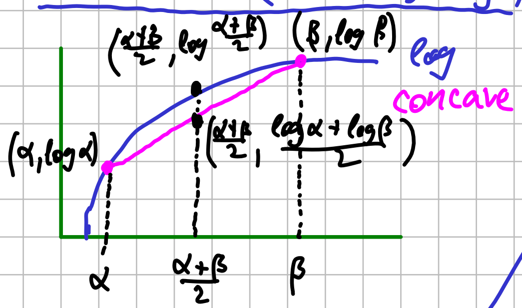
$r_0, r_1, \dots, r_t$

then amort. cost of i-th step $\le 3(r_i(x) - r_{i-1}(x)) \underbrace{+1}_{\text{only for zig}}$

→ total am. cost of Splay

$$\le \underbrace{\left(\sum_{i=1}^{t} 3(r_i(x) - r_{i-1}(x))\right)}_{3(r_t(x) - r_0(x))} + 1$$

Lemma M (means of logs): $\forall \alpha, \beta > 0 \quad \log \frac{\alpha+\beta}{2} \ge \frac{\log \alpha + \log \beta}{2}$



Proof: Consider AM-GM inequality

$$\forall \alpha, \beta > 0 \quad \sqrt{\alpha\beta} \le \frac{\alpha+\beta}{2}$$

$\xrightarrow{\log}$

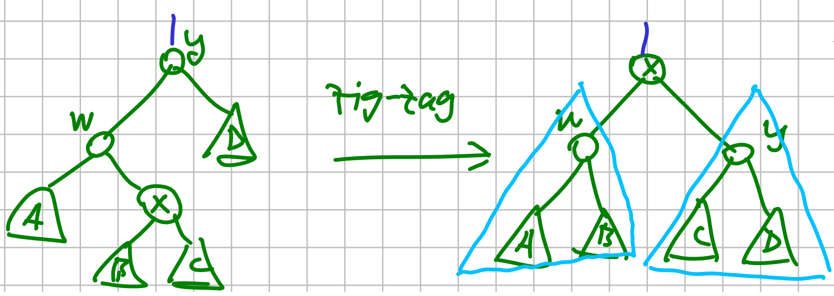
$$\frac{1}{2}(\log \alpha + \log \beta) \le \log \frac{\alpha+\beta}{2}$$

$$2 \log \frac{\alpha+\beta}{2} \ge \log \alpha + \log \beta$$

Lemma S (sum of logs)

$$2 \log(\alpha+\beta) - 2 \ge \log \alpha + \log \beta$$

Case 1 zig-zag



$T(w) \supseteq T(x)$

$s(w) \ge s(x)$

$r(w) \ge r(x)$

$-r(w) \le -r(x)$

real cost = 2

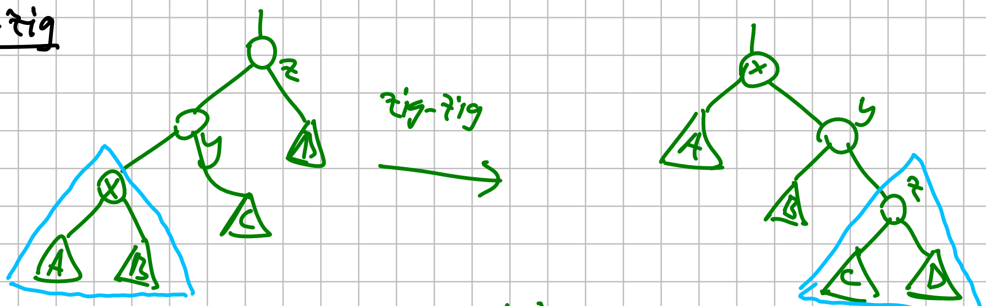
we want to show this $\leq 3(r'(x) - r(x))$

$$\text{am. cost} = 2 + \Delta\Phi = 2 + r'(w) + r'(x) + r'(y) \underbrace{- r(w)}_{\leq -r(x)} - r(x) \underbrace{- r(y)}_{\leq -r(x)}$$

$$\log s'(w) + \log s'(y) \overset{\text{Lemma 5}}{\leq} 2\log(\underbrace{s'(w) + s'(y)}_{\leq s'(x)}) - 2 \leq 2r'(x) - 2$$

$$\leq 3r'(x) - 3r(x)$$

Case 2: zig-zig



real cost = 2

we want: $\leq 3r'(x) - 3r(x)$

$$\text{am. cost} = 2 + \Delta\Phi = 2 + r'(x) + \underbrace{r'(y)}_{\leq r'(x)} + \underbrace{r'(z)}_{\leq 2r'(x) - r(x) - 2} - r(x) \underbrace{- r(y)}_{\leq -r(x)} \underbrace{- r(z)}_{= -r'(x)}$$

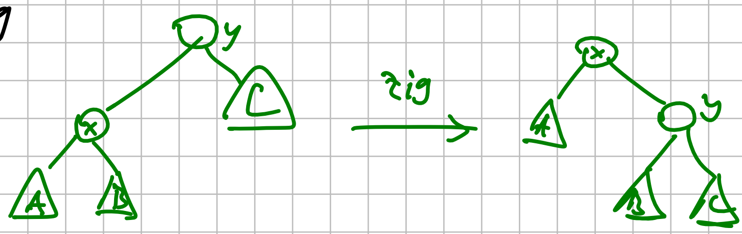
Use Lemma 5:

$$r(x) + r'(z) = \log s(x) + \log s'(z) \leq 2\log(\underbrace{s(x) + s'(z)}_{\leq s'(x)}) - 2 \leq 2r'(x) - 2$$

$$\Rightarrow r'(z) \leq 2r'(x) - r(x) - 2$$

$$\leq 3r'(x) - 3r(x).$$

Case 3: zig



real cost = 1

$$\text{am. cost} = 1 + \Delta\Phi = 1 + r'(x) + \underbrace{r'(y)}_{\leq r'(x)} - r(x) \underbrace{- r(y)}_{\leq -r(x)}$$

$$\leq 1 + 2r'(x) - 2r(x) \leq 1 + 3r'(x) - 3r(x)$$

because $r'(x) - r(x) \geq 0$

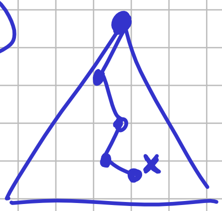
Recall: Total real cost of a sequence of ops $\leq$ Total amort. cost of the seq. $+ \underbrace{\Phi_{start}}_{\leq n\log n} - \underbrace{\Phi_{final}}_{\leq 0}$

m operations Splay on n-node tree; total amort. cost $O(m \cdot \log n)$

↳ __Theorem:__ Total cost of m Splays on a n-node tree is $O\big((m+n)\log n\big)$.

Splay tree as dynamic search tree

Find



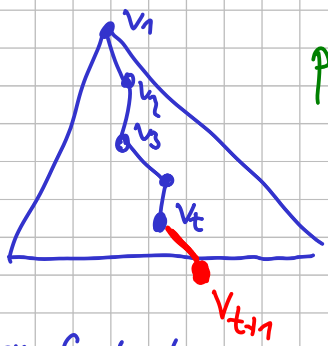
When we find x, we splay it

__Trick:__ Account traversing the path on splaying.

→ am. cost of Find is O(am. cost of Splay) is $O(\log n)$.

! Unsuccessful Find: Splay the stopping node

Insert



then Splay(v_{t+1})

problem: attaching the leaf v_{t+1} has real cost ≈ 0, but $\Delta\Phi$ could be huge!

$$\Delta\Phi = \underbrace{r'(v_{t+1})}_{\substack{=0 \\ \text{(leaf)}}} + \left(\sum_{i=1}^{t} \underbrace{r'(v_i)}_{\log s'(v_i)} - r(v_i)\right)$$

$$\log s'(v_i) = \log(\underbrace{s(v_i)+1}_{\le s(v_{i-1})}) \le r(v_{i-1})$$

$$\le r'(v_1) + \cancel{r(v_1)} + \cancel{r(v_2)} + \ldots + \cancel{r(v_{t-1})} - \cancel{r(v_1)} - \cancel{r(v_2)} - \ldots - \cancel{r(v_{t-1})} - r(v_t)$$

$$\le r'(v_1) \le \log n.$$

Delete → practicals

__Idea:__ Same as ordinary BST & Splay the lowest node visited.

Theorem on Static Optimality

↳ next lecture → much more in Data Structures 2

applications