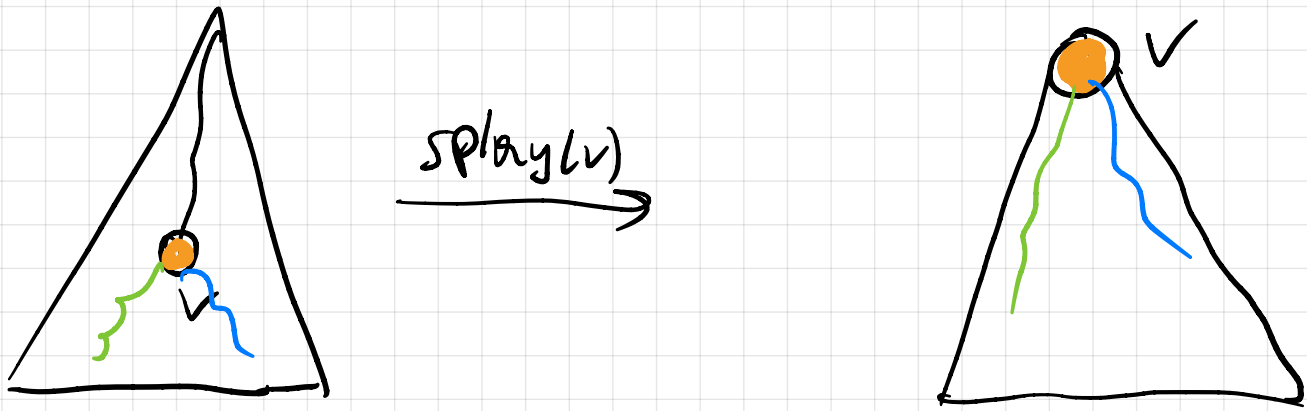
Splay Tree

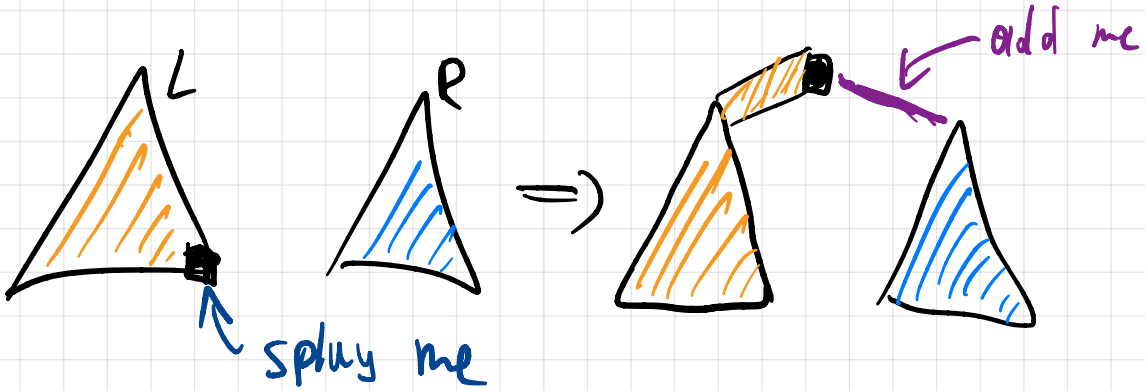
[notes not fully complete... :(]

def: Splay operation

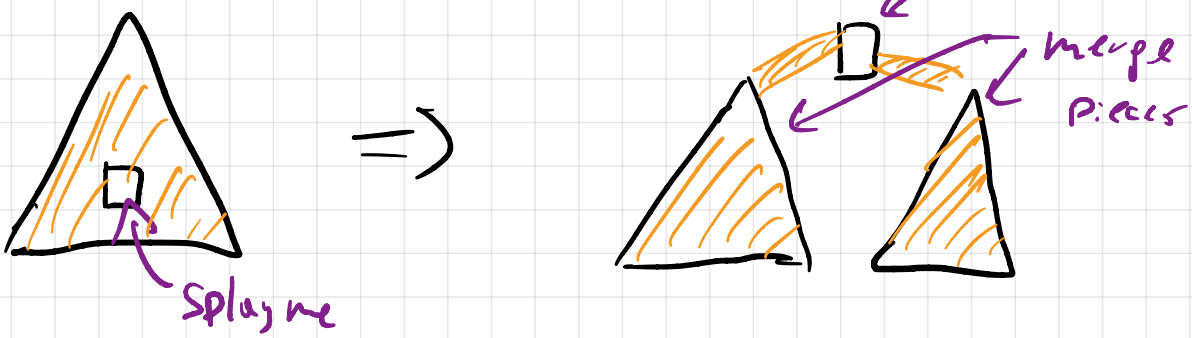


Note: if you can splay you can do anything:*

merge:



delete



*: T&C APPLY

split & insert too (more on that later)

Note: Splay(v) works in $O(\text{depth}_v)$

However, depth_v can be linear. ???!

Introducing: ★ Amortized Analysis ★

φ - Potential Function

a numerical measure of Data Structure State

φ_n $(n \geq 0)$ — State after n operations

Require $\varphi_n \geq 0$ $\forall n$,

$\varphi_0 = 0$

Let t_n — real time of operation n

Let $a_n = t_n + \varphi_n - \varphi_{n-1}$ — amortized time of operation n

$$\varphi_0 \xrightarrow{t_1} \varphi_1 \xrightarrow{t_2} \varphi_2 \dots$$

Note: $\sum_{k=1}^{n} a_k = \sum_{k=1}^{n} t_k + (\varphi_n - \varphi_0)$, $\varphi_n - \varphi_0 \geq 0$

Statement $\sum_{k=1}^{n} a_k \geq \sum_{k=1}^{n} t_k$

i.e. we can use a_k to think of operation times, and we won't underestimate total work needed

(Another way to think of φ is a coins purse, on a good day you put money into it, $\varphi_n - \varphi_{n-1} > 0$, on worse days you use it to pay for things)

C++ Vector Example: increase φ by 1 every push-back. If Reallocation happens, $\varphi : n \to 0$.

back to Splay.

Let $w_v > 0$ be the element's weight

Normally, $w_v = 1$.

Let $\text{size}_v = \sum_{u \in \text{Subtree}(v)} w_u$

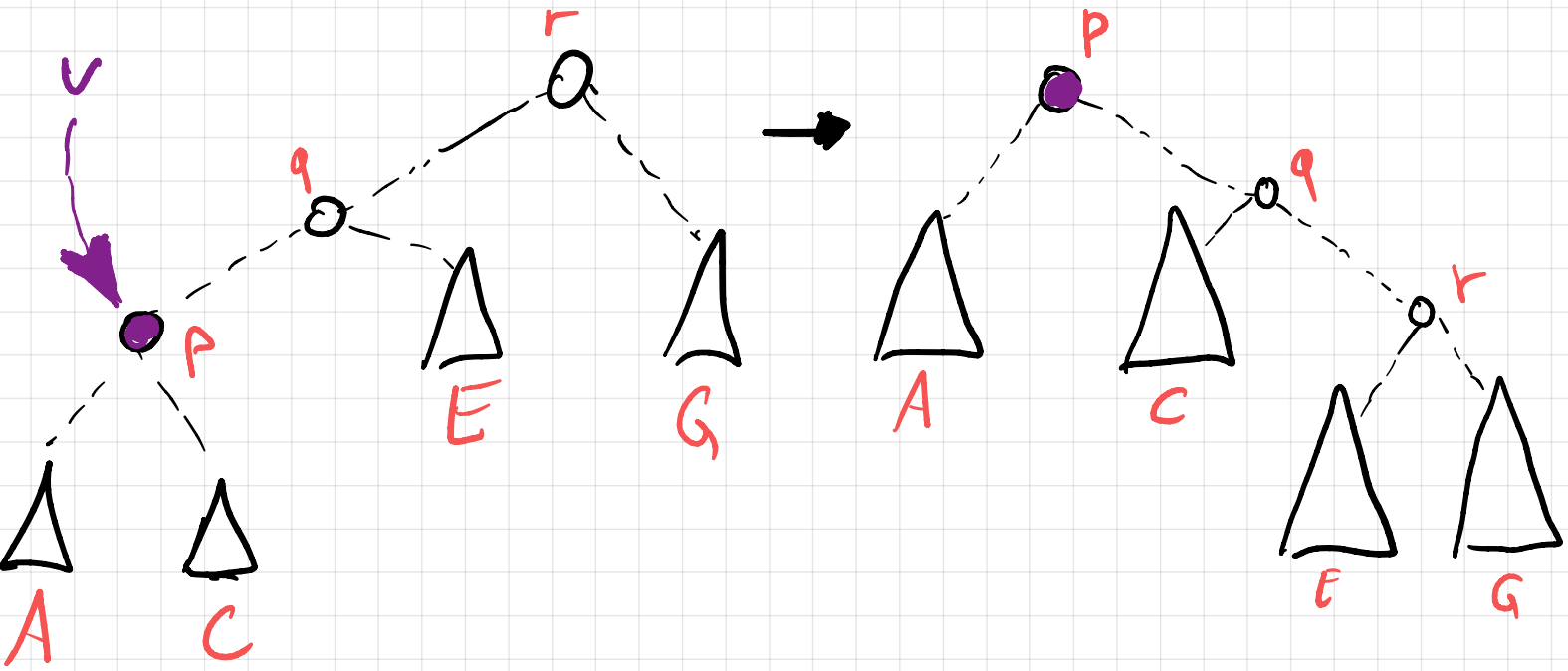
Let $\varphi = \sum_v \log \text{size}_v$

Now consider Splay(v) operation:

We should use specific operations to

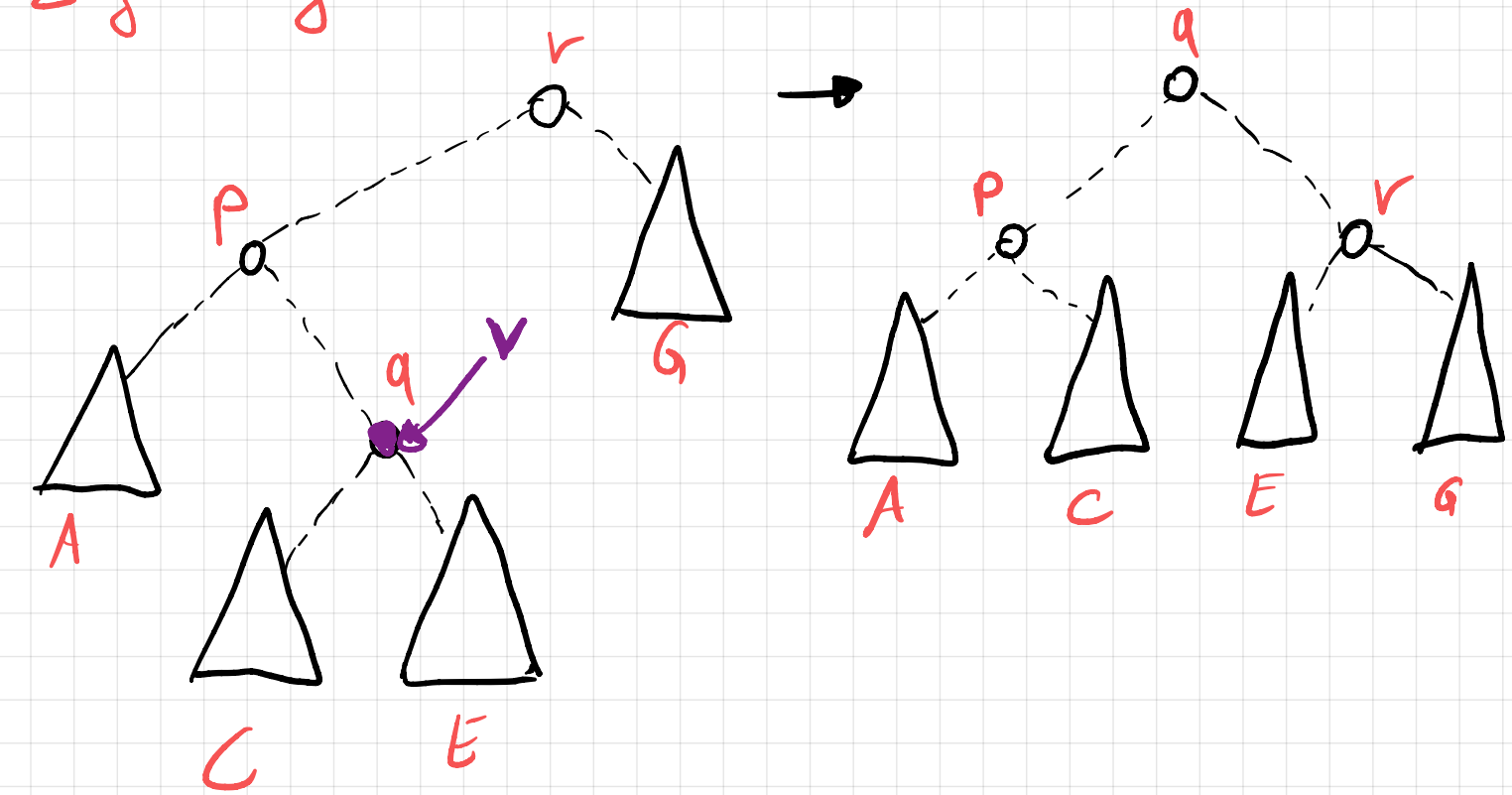
move v to top:

Zig Zig

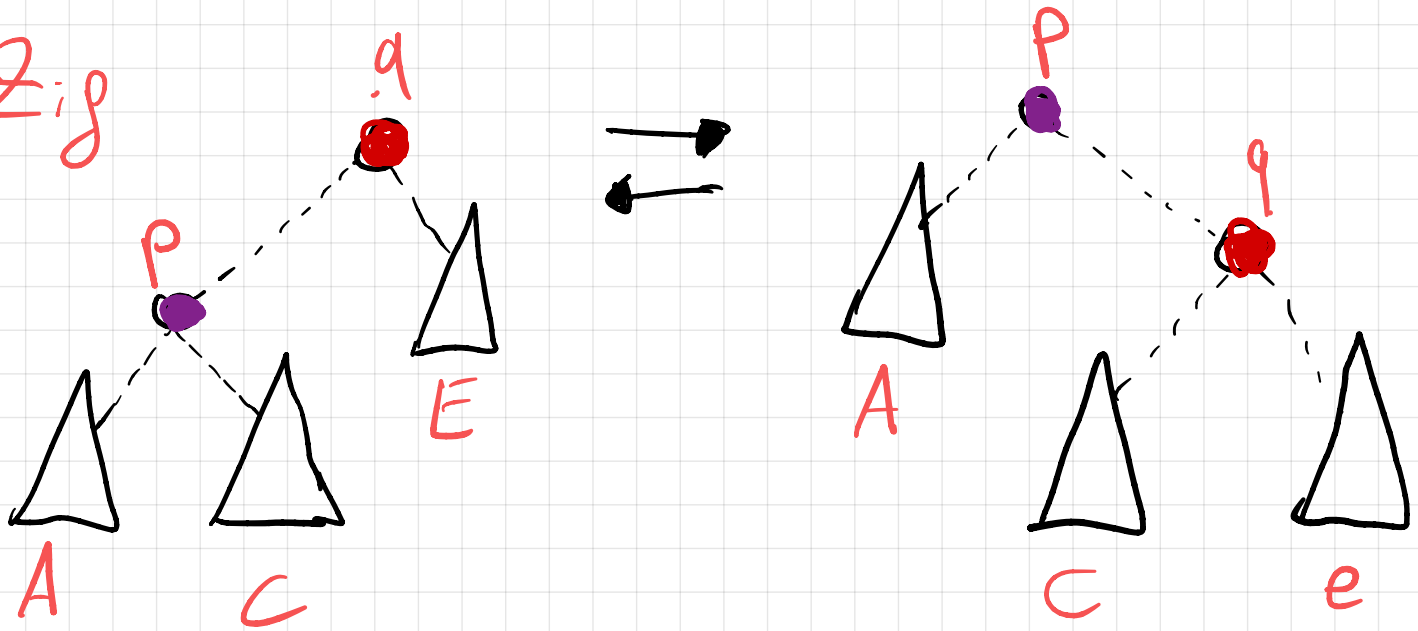


A<p<C<q<E<r<G

Zig Zag

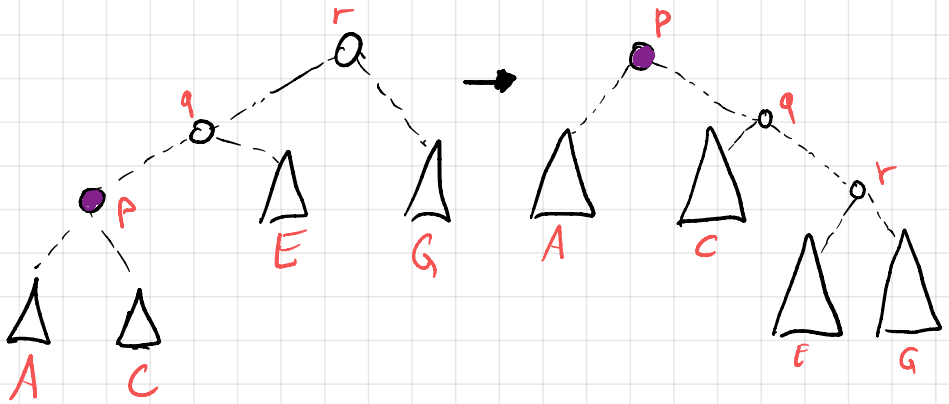


Zig



Potential Analyser

Zig-Zig:



$$\Delta\varphi = \varphi_{new} - \varphi_{old} =$$

$$= \log(Size_p^{new}) - \log(Size_p^{old}) \quad + p_{old}$$

$$+ \log(Size_q^{new}) - \log(Size_q^{old}) \quad + p_{old}$$

$$+ \log(Size_r^{new}) - \log(Size_r^{old}) \quad + p_{old}$$

$$= \log(w_q + w_C + w_r + w_E + w_G) + \log(w_r + w_E + w_G)$$

$$- \log(w_q + w_p + w_A + w_C + w_E) - \log(w_{p+} w_A + w_C)$$

Claim: $\Delta\Psi \leq 3\left(\log Size_p^{new} - \log Size_p^{old}\right)$

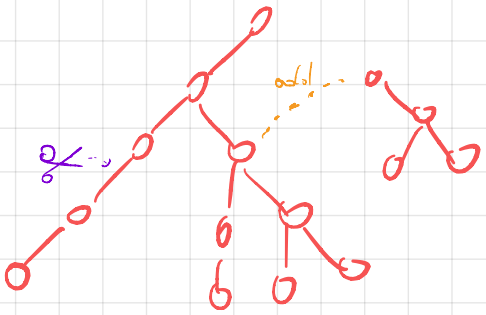
Static finger Theorem

Suppose Splay stores keys $\{1, 2, \ldots, n\}$ and there are find requests for elements $a_1, a_2, \ldots, a_m$.

Then Splay works in

$$O(m + n \log n + \sum \log(|a_i - t| + 1))$$

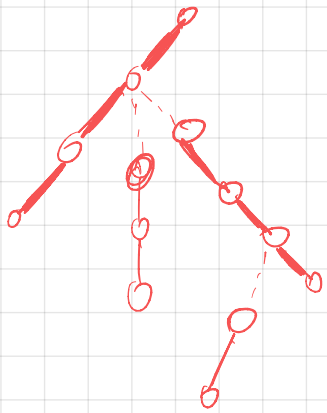
Link - Cut



maintain forest

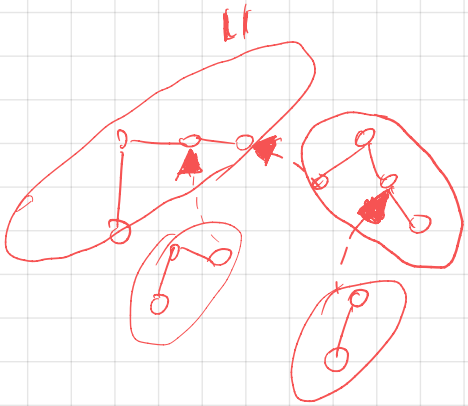
add edges, cut edges, path queries

methodology:



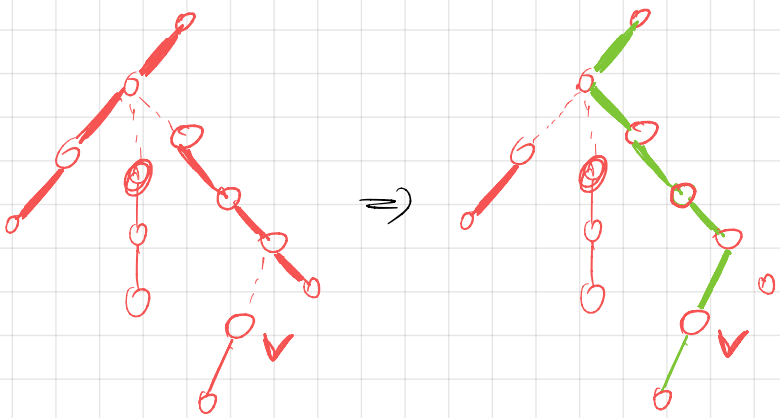
vertical paths

stored in Binary Search Trees



Node*

Expose(v)



Change Root

Expose(v)

Cut Below v

Reverse

Add Edge (v, u)

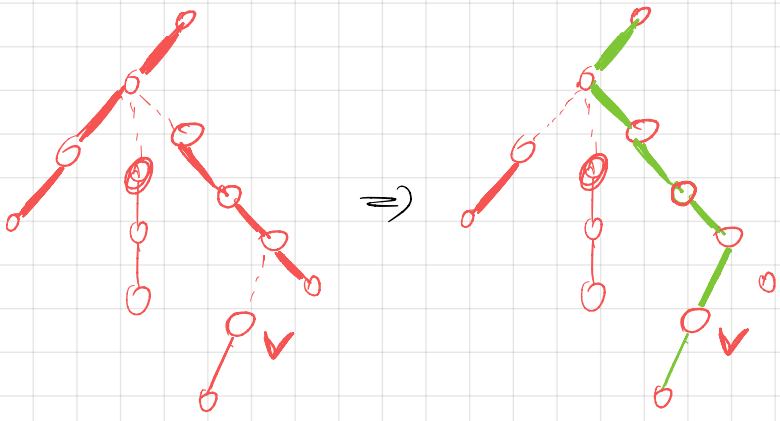
ChangeRoot(u)

u → tree parent = v

Len (v, u)

LCA(v, u) — exercise

Expose Complexity



(t = 3 here)

Let t be the number of hops we had to do.

Then $time_{expose} = O(t \cdot \log n)$ ← BST operation complexity

⚠ t can be large.

⇒ Amortization is needed

Claim: $\sum_{i=1}^{m} t_i = O(m \log n)$

Def (Similar to HLD/DSU analysis)

– heavy if $\frac{\text{Tree Size}_v}{\text{Tree Size}_p} \geq \frac{1}{2}$

light otherwise

$$t_i = \text{heavy}_i + \underbrace{\text{light}_i}_{\text{heavy \& light hops. \& starting path}} + 1$$

Note $\text{light}_i \leq \log_2 n$ since going up light edge doubles sub tree size.

Let $\varphi := \#\text{heavy edges}$ not covered by paths decomp

$$\text{heavy}_i + \Delta\varphi_i \leq \log_2 n$$

(all heavy_i edges get covered which results in appropriate $\Delta\varphi$ reduction.

$\text{light}_i \leq \log_2 n$ light edges also get covered, potentially uncovering $\leq \text{light}_i$ heavy edges, increasing $\Delta\varphi$ by $\leq \log_2 n$)

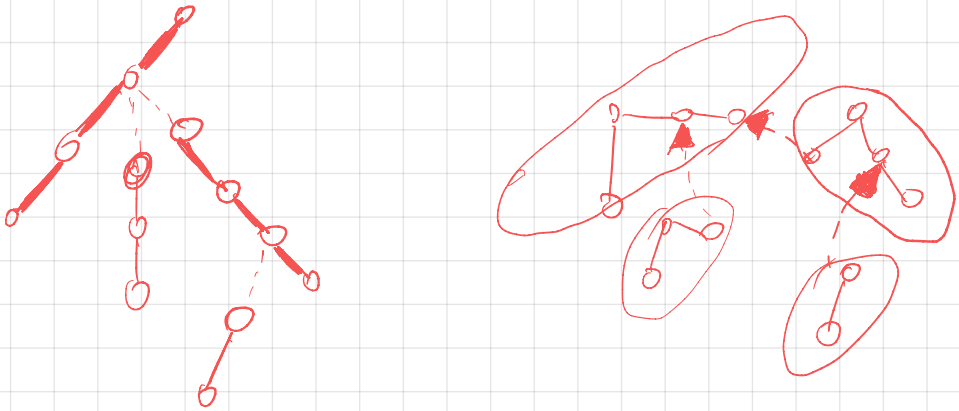
$$\Rightarrow \sum \text{heavy}_i \leq m \log_2 n$$

$$\Rightarrow \sum t_i \leq m \log_2 n + m \log_2 n + m$$

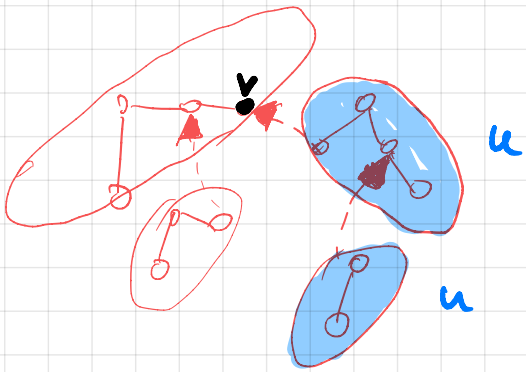
□

Splay Trees ♥ Link Cut Trees

Recall in Splay, $\text{time}_{BST} \leq 3(\log \text{Size}_v^{new} - \log \text{Size}_v^{old}) + 1$



Let $w_v = 1 + \sum_{u \text{ - directly or indirectly hangs by } v} w_u$



$$\text{time}_{\text{expose}_v} = \sum_{i=1}^{t} \text{time}_{BST_i}$$

$$\leq \sum_{i=1}^{t} \underbrace{3(\log \text{Size}_{v_i}^{new} - \log \text{Size}_{v_i}^{old})}_{\text{telescopic sum}} + 1$$

$$\leq 3 \log_2 n + t$$

recall $\sum_{i=1}^{m} t = O(m \log n)$

$$a_{\text{expose}_v} = O(\log_2 n). \quad \square$$