Balanced binary search trees

Goal: $O(\log n)$ time per op

C++ STL red-black tree complicated

treap Aragon Seidel '90

binary tree: every node has
a key and a priority

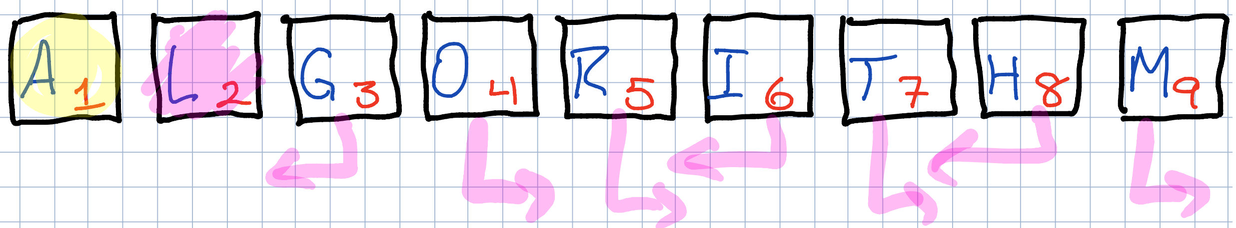
Binary search tree min-heap

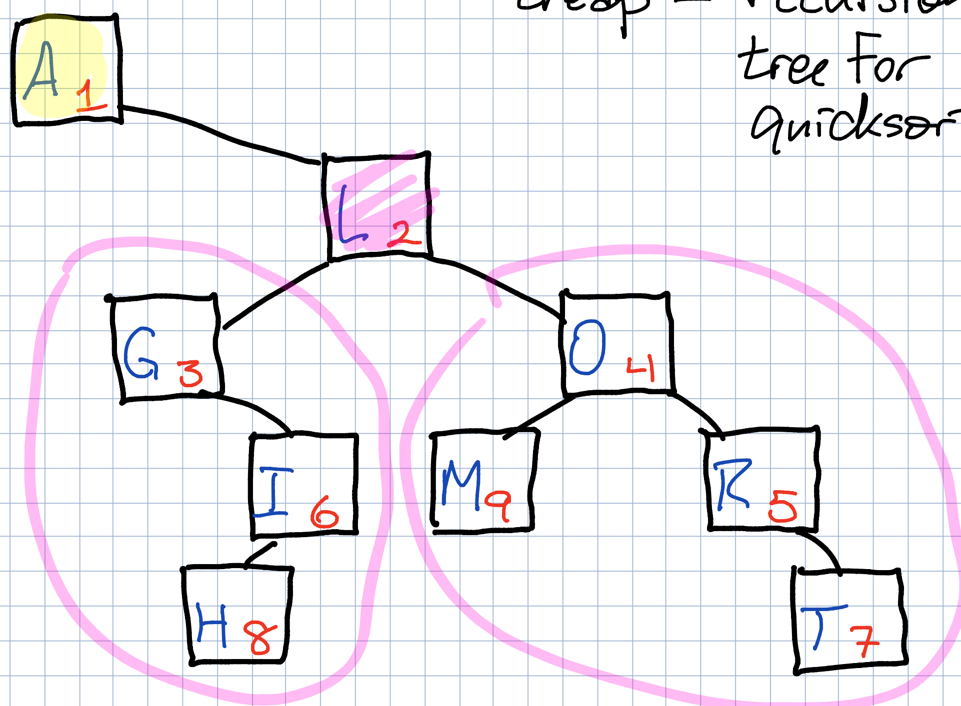
Unique for any given keys + priorities

Given an arbitrary set of keys
find priorities s.t. treap is balanced

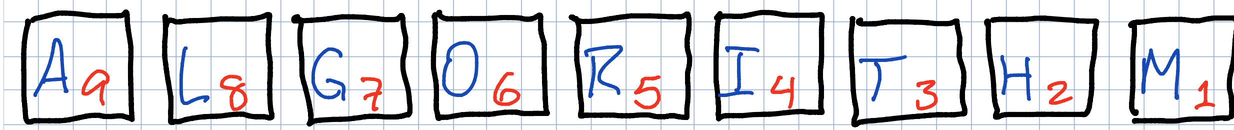
Priorities are random
uniform real numbers in $[0,1]$

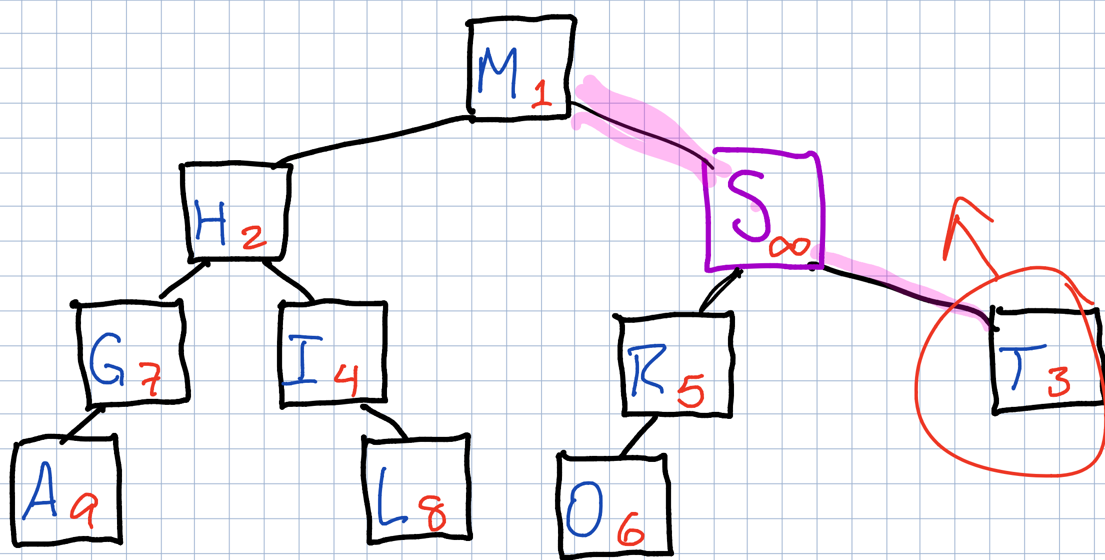
unique with prob. 1.



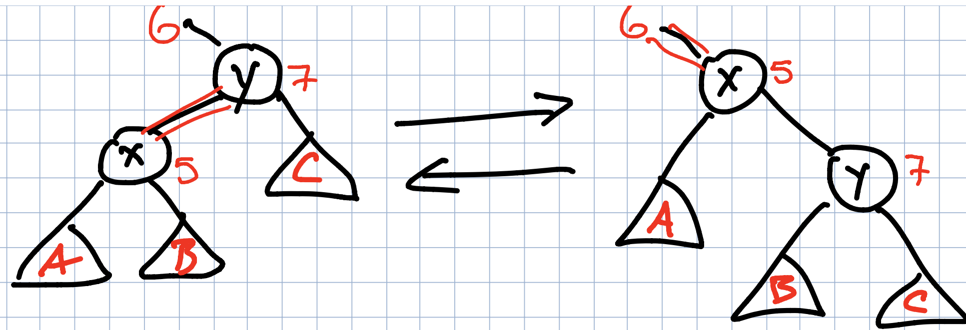


treap = recursion tree for quicksort!






```
Insert(x):
    Assign x a random priority
    BST insert(x) ⇐ depth
    while pri(x) < pri(parent(x))
        rotate(x)
Delete(x):   = Insert backwards in time
    pri(x) ← ∞
    while (x is not a leaf)
        if pri(left(x)) < pri(right(x))
            rotate(left(x))
        else
```



Time for each op is depth of some node

Claim: For every key x, if all priorities are random, $E[\text{depth}(x)] = O(\log n)$

$$E[\text{depth}] = E[\#\text{ proper ancestors}]$$

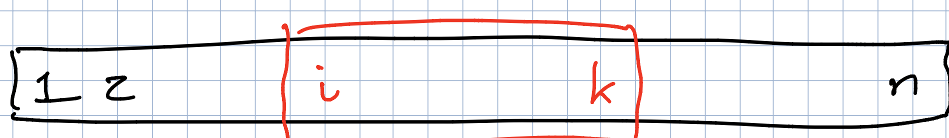
$$= \sum_v \Pr[v \text{ is a proper ancestor}]$$

node i

$[i \uparrow k] = 1$ if node with i^{th} smallest key is proper ancestor of the node with k^{th} smallest key

node k

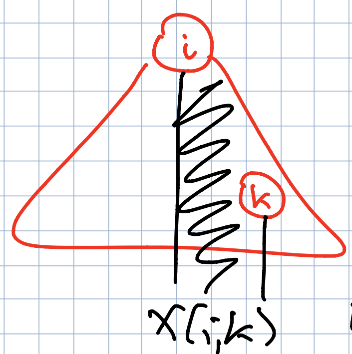
$\Pr[i \uparrow k] = \Pr[$randomized quicksort compares $i:k$ when i is pivot$]$



$X(i,k) = \{i, i+1, \dots, k\}$ or $\{k, k+1, \dots i\}$

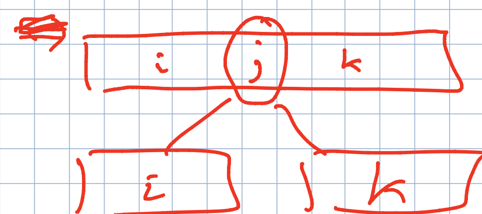
<u>Claim:</u> $[i \uparrow k] \Leftrightarrow$ smallest priority in $X(i,k)$ is node i

<u>Proof:</u> $\Longrightarrow$



$\square$

$[\Longleftarrow]$ smallest in $X(i,k)$ is node j



$\square$

$$\Pr[i \uparrow k] = \Pr[i \text{ is smallest in } X(i,k)]$$

$$= \frac{1}{|X(i,k)|} = \frac{1}{|k-i|+1}$$

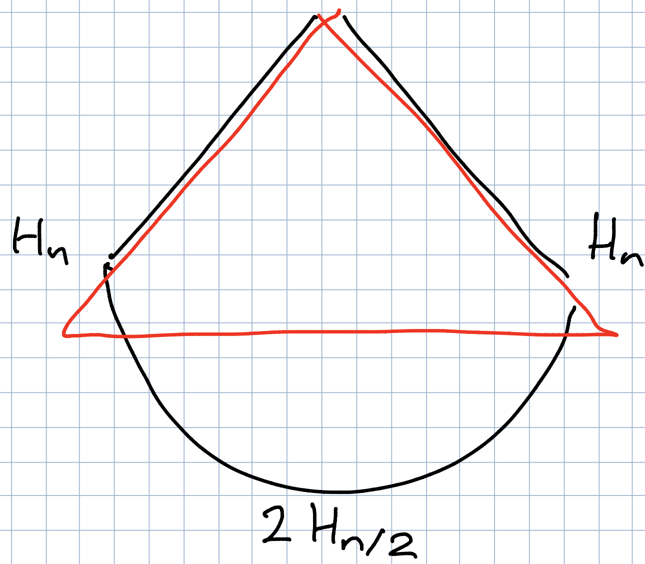
$$E[\text{depth}(i)] = \sum_k \Pr[k \uparrow i]$$

$$= \sum_{k<i} \frac{1}{i-k+1} + \sum_{k>i} \frac{1}{k-i+1}$$

$$\le 2H_n$$

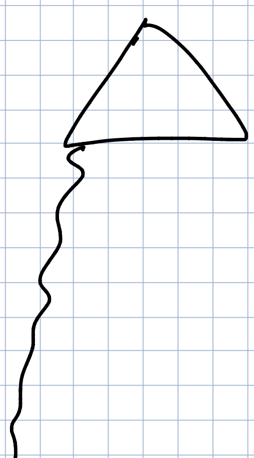
$$\le 2\ln(n+1)$$

$H_k + H_{n-k+1} - 1$



for all x $E[\text{depth}(x)] \le O(\log n)$

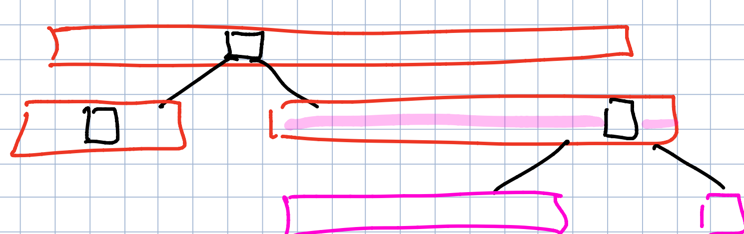
$E[\underline{\max} \text{ depth}] = \quad O(\log n)$



$\Pr[\text{max depth} > 4 \ln n] = ?$

level is <u>bad</u> if biggest subproblem

is split badly



$\Pr[\text{level is bad}] = 1/2$

max # good levels $\le \log_{4/3} n$

$\Pr[\#\ \underline{\text{bad}}\ \text{levels} \geq 3\log n]$

$= \Pr(\text{Flip coin } 4\log n \text{ times}$ $\qquad 2^{-3\log n}$

$\qquad \Rightarrow 3\log n\ \underline{\text{tails}}]$ $\qquad \approx n^{-3}$

Tail inequality : $\boxed{\Pr[X > \alpha E[X]] \leq ?}$