15-750
1/20/16
V2

Binary Search Trees

Data Structure Dictionary

S is an ordered set.

1) Search$(K, S) \equiv K \in S$?
2) Insert$(K, S) \equiv$
3) Delete(K, S)

Note If 1), 2), 3) are the Design Requirement then use a Hash table

4) Range$(K, K', S) \equiv |\{K'' \in S \mid K \le K'' \le K'\}|$

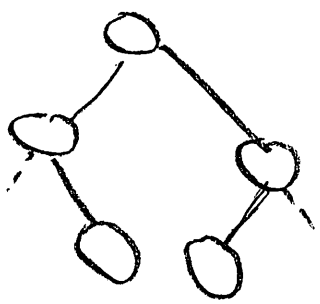
If 1), …, 4) use BST

<u>Def</u> A tree T is BST for keys S if:

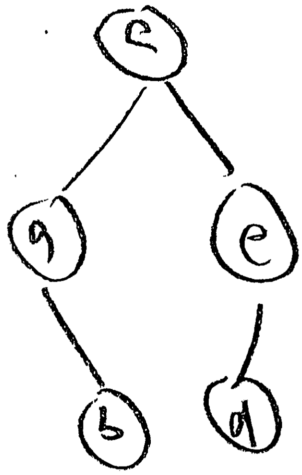
1) T is an ordered binary tree with |S| nodes.
2) Each node stores a key.
3) Keys are in inorder

---

eg $S = \{a, b, c, d, e\}$

T =  $\Rightarrow$

BST



T is <u>balanced</u> if $\text{Maxdepth}(T) = O(\log n)$

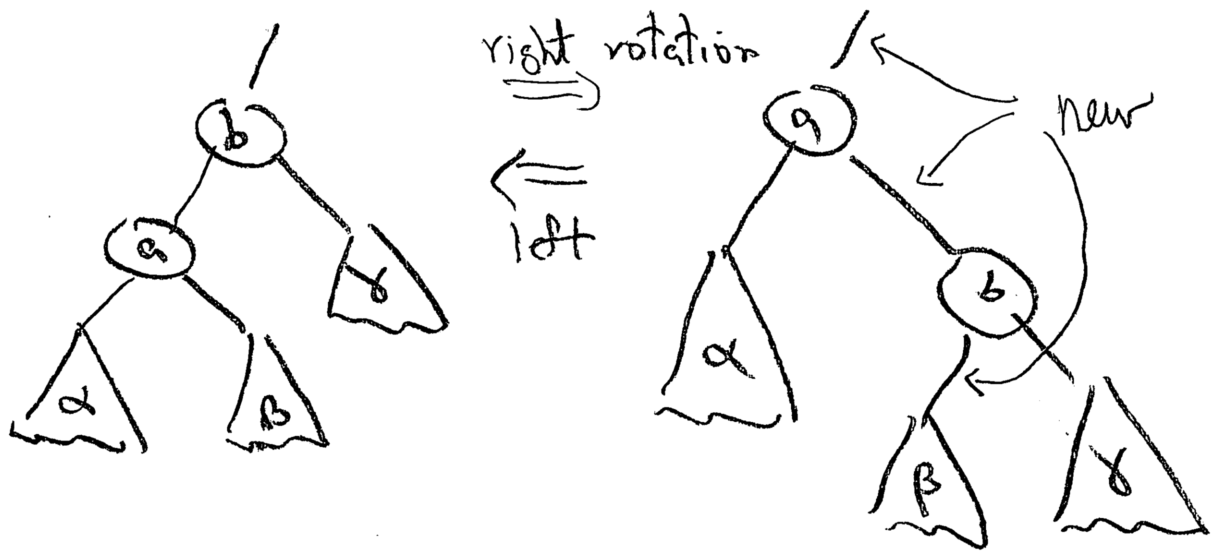
Types of Balanced BST's

Always Balanced – AVL, 2-3-4, RB, B-Trees

Randomized – skip-lists, Treaps ≡ tree-heaps

Amortized – Splay Trees

All these use the Rotation



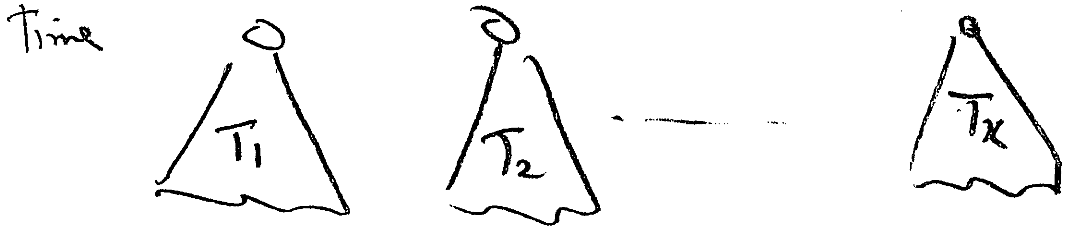
To show: inorder is preserved

Note: Subtrees α, β, γ unchanged!

Applications

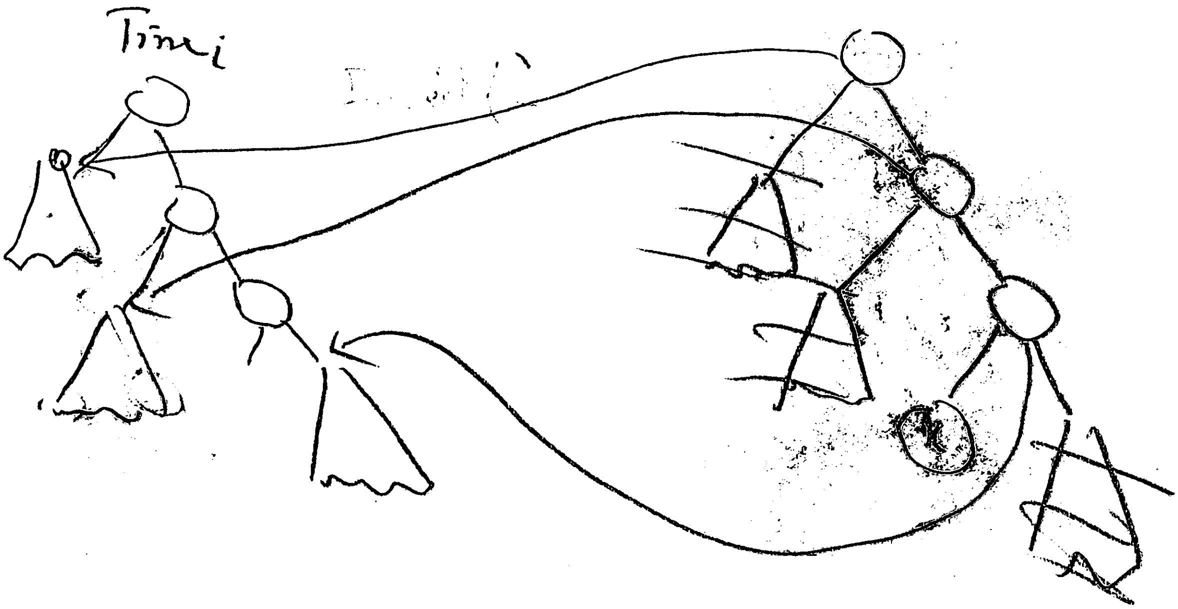
Persistance (Add 5) undo last op)

First Idea Keep a tree for each time



$O(n^2)$ spaces.

Idea 2 store only diffences Insert(x)

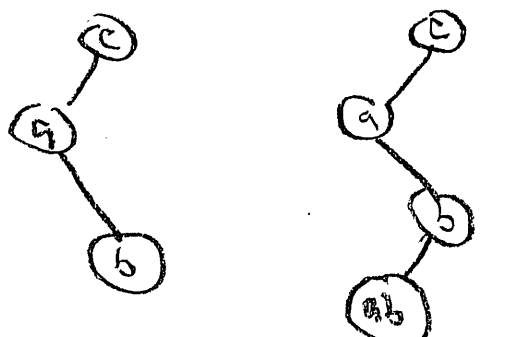


$O(\log n)$ space per new tree.

Vanilla-Insert (K T)

1) Find empty leaf node

2) add k to leaf



T   VI(ab, T)

QS(A)   Bad Quick Sort

1) Pick first a in A

2) Split S into S<a, a, a<L

3) Return QS(S), a, QS(L)

Eager

Lazy

Vanilla-BST(A, T)   VBST

1) Extract first a from A

2) Return VBST(A, VI(a, T))

Note QS & V-BST do exactly the same comparisons but in different order!

Treaps (Tree-Heaps)

Keys $\equiv 1, \cdots, n$

Priorities $\equiv p(k)$ $\quad p(k) \neq p(l)$ for $k \neq l$

$T \equiv$ tree with a key at each node.

Def T is in heap order if $\forall x \in T$ x not root

$$p(\text{parent}(x)) < p(x)$$

Lemma A heap order BST exists and is unique.

pf Vanilla-insert in priority order.

Random Treap

Insert(K) ≡ 1) insert K into a leaf (VI(K,T))
2) pick random p(K)
3) rotate K up until in heap order

Delete(K) ≡ 1) Rotate K to a leaf
by picking highest priority child
2) remove K.

Correctness:

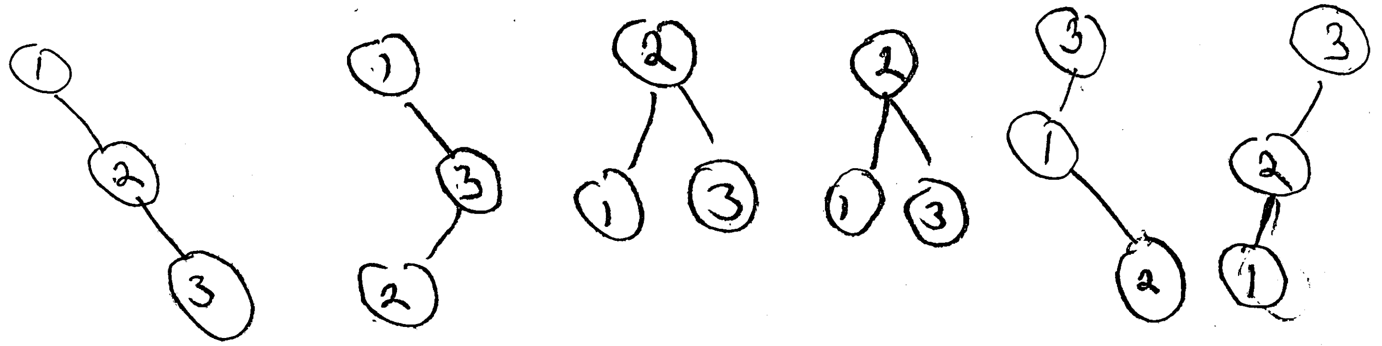
Expected Cost for Treaps

Goal: Determine expected # of comparisons to search$(m.5, K) \equiv S(m.5)$ over all treaps.

eg $K = \{1,2,3\}$ & search$\{2.5, K\}$

Treaps insert orders $n!$ 4^n trees

(123) (132) (213) (231) (312) (321)



3 3 2 2 3 2

$$S(2.5) = \frac{15}{6} = 2\tfrac{1}{2}$$

Note for random BST $\frac{13}{5} = 2\tfrac{3}{5}$

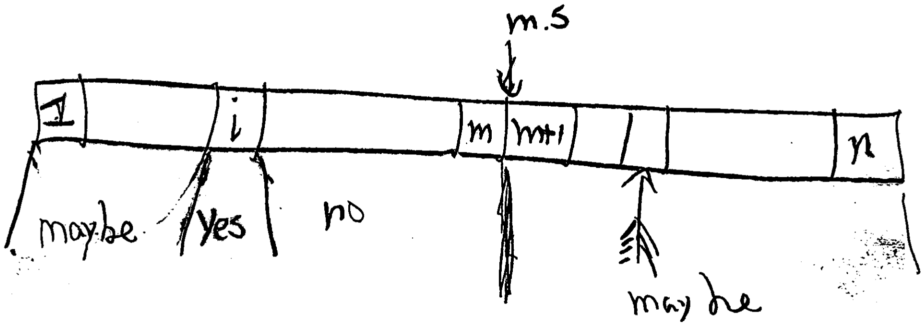
Keys $\{1, \dots, n\}$ in Treap

$C(i,m)$ = Event [i is compared to m in Search(m)]

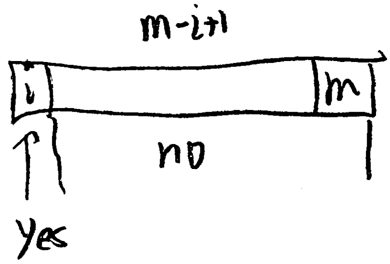
<u>Claim</u> $\text{Prob}[C(i, m.5)] = \frac{1}{m-i+1}$ if $i \le m$

$= \frac{1}{i-m}$ if $i > m$

Treap construction as dart game



informal argument: WLOG board is



$\text{Prob}[C(i, m.5)] = \frac{1}{m-i+1}$

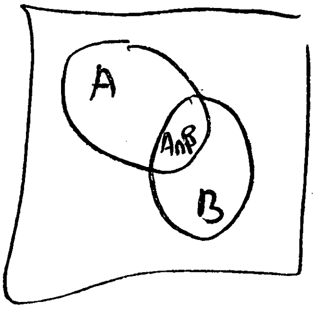
Formal Argument (Using Law of Total Probability)

B_j = Event [jth dart is first to land in $[i,\dots,m]$]

$B_k \cap B_j = \emptyset$ if $k \neq j$ & $\Pr\left(\bigcup_{j=1}^{\infty} B_j\right) = 1$ (*)

Conditional Probability: Let A, B be events

$$\Pr[A|B] = \frac{\Pr[A\cap B]}{\Pr[B]}$$



$i \le m$

$$\Pr[\underbrace{C(i,m,5)}_{=A}] = \sum_{j=1}^{\infty} \Pr[B_j]\Pr[A|B_j]$$ true by (*)

note: $\Pr[A|B_j] = \frac{1}{m-i+1}$

$$\Pr[A] = \sum_j \Pr[B_j]\left(\frac{1}{m-i+1}\right) = \frac{1}{m-i+1}\sum_{j=1}^{\infty}\Pr[B_j] = \frac{1}{m-i+1}$$

QED

$$S(m.5) = \#\text{ comparison searching } m.5$$

$$= \sum_{i=1}^{n} C(i, m.5)$$

$$E(S(m.5)) = \sum E(C(i,m.5)) = \{\text{Prob}[C(i,m.5)]\}$$

$$= \sum_{1 \le i \le m} \frac{1}{m-i+1} + \sum_{i > m}^{n} \frac{1}{i-m} \le 2\sum_{i=1}^{n} \frac{1}{i} = 2H_n$$

$$= 2\ln n + O(1)$$

Expect # of Comparisons for

Insert, Search, Delete $= O(\log n)$

Counting Rotations

2 Cases: Insert & Delete

Delete: Move-to-Leaf

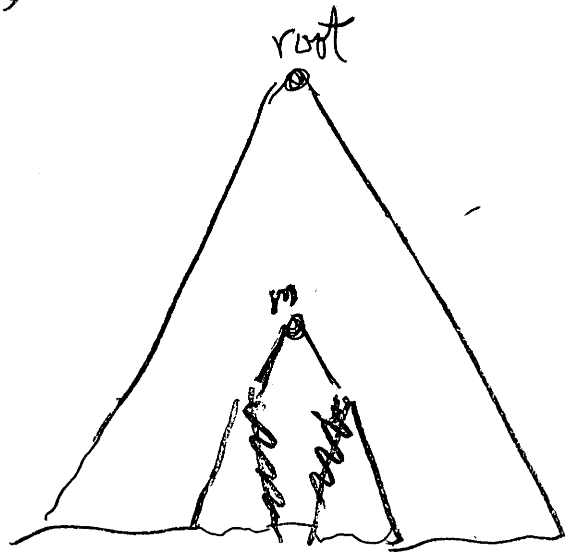
Thm.: Expect #rotations < 2

Claim Pivots with m are

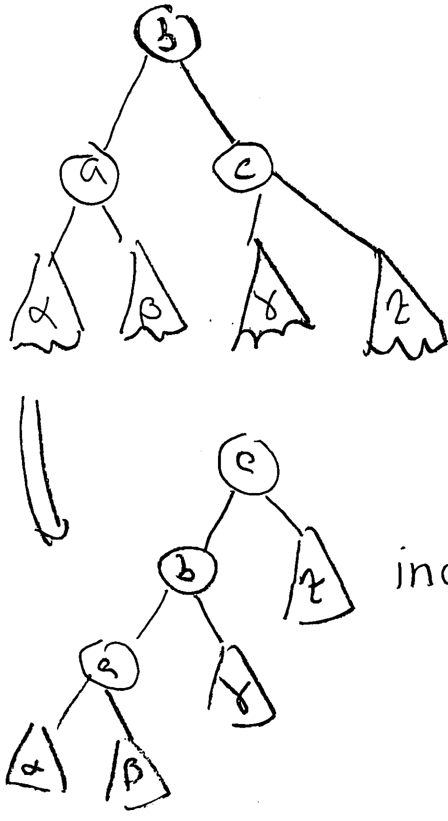
1) Right-most nodes in left subtree of m

2) Left " " right " "

eg



pf



induct!

Def Random varibles D_i & D_i'

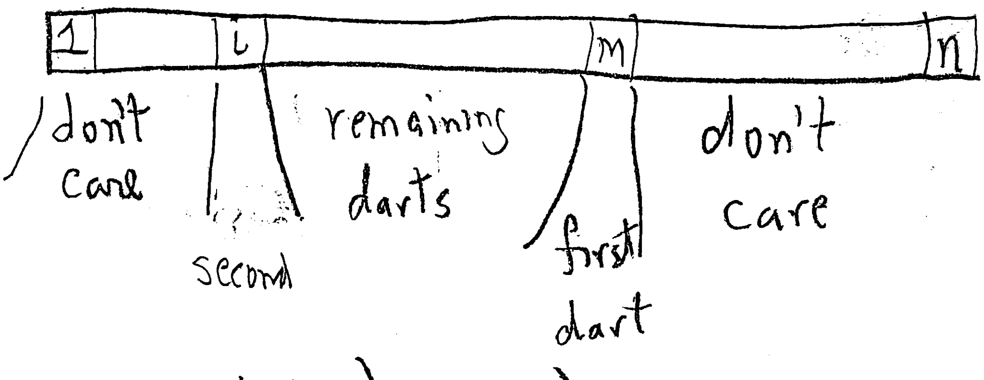
$$D_i = \begin{cases} 1 & \text{if } i \text{ is a right-most node in left subtree of } m \\ 0 & \text{o.w} \end{cases}$$

$$D_i' = \begin{cases} 1 & \text{if } i \text{ is a left-most node in right subtree of } m \\ 0 & \text{o.w.} \end{cases}$$

Claim $\Pr[D_i = 1] = \left(\frac{1}{m-i+1}\right)\left(\frac{1}{m-i}\right)$

pf Dart Game (Informal)

Board for D_i



$$\Pr[D_i = 1] = \left(\frac{1}{m-i+1}\right)\left(\frac{1}{m-i}\right)$$

$$\Pr[D_i' = 1] = \left(\frac{1}{i-m+1}\right)\left(\frac{1}{i-m}\right)$$

<u>Def</u> R_m = # rotations in move-to-leaf of m

$$E(R_m) = \sum_{i<m} E(D_i) + \sum_{i>m} E(D_i')$$

$$\leq \sum_{i=1}^{m-1} \frac{1}{(m-i+1)(m-i)} + \sum_{i=m+1}^{n} \frac{1}{(i-m+1)(i-m)}$$

$$\leq \underbrace{\sum_{i=1}^{m-1} \frac{1}{(i+1)\,i}} + \sum_{i=1}^{n-m} \frac{1}{(i+1)\,i}$$

<u>note</u> $\frac{1}{(i+1)\,i} = \frac{1}{i} - \frac{1}{i+1}$

LH Term $= \sum_{i=1}^{m-1} \frac{1}{i} - \sum_{i=2}^{m} \frac{1}{i} = 1 - \frac{1}{m} < 1$

$E(R_m) < 2$