Data Structure  Dictionary

S is an ordered set.

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

---

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

4) Range \((k, k', S) \equiv \{ k'' \in S \mid k \leq k'' \leq k' \}\)

If 1), ..., 4) use BST
**Def** A tree $T$ in 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.

**Eq** $S = \{a, b, e, d, e\}$

$T =$ \[
\begin{array}{c}
\circ \\
\circ \\
\circ \\
\end{array}
\]  $\Rightarrow$  \[
\begin{array}{c}
\circ \\
\circ \\
\circ \\
\circ \\
\circ \\
\end{array}
\]

$T$ is balanced if $\text{max depth}(T) = O(\log n)$
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
Applications

Persistence (undo)

First Idea  Keep a tree for each time

Time

$T_1$

$T_2$

$T_X$

$O(n^3)$ space.

Idea 2  store only differences  Insert $(x)$

$O(\log n)$ space per new tree.
Vanilla - Insert $(k,T)$

1) Find empty leaf node
2) Add $k$ to leaf

$QS(A)$  Bad Quick Sort

1) Pick first $a$ in $A$
2) Split $S$ into $S_{<a}$, $a$, $S_{>a}$
3) Return $QS(S_{<a}), a, QS(S_{>a})$

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

Keys \in 1, \ldots, n

Priorities \equiv p(k) \quad p(k) \neq p(l) \text { for } k \neq l

T = \text{tree with a key at each node.}

\textbf{Def} T \text{ is in heap order if } \forall x \in T \text{ not root } p(\text{parent}(x)) < p(x)

\underline{Lemma} A heap order BST exists and is unique.

\textbf{Pr} Vanilla-insert in priority order.
Random Treap

Insert \((k)\) =
1) insert \(k\) into a leaf
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.s, K) = S(m.s)

eg K = {1,2,3) & search (2.5, K)

Treaps insert order

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

\[
\begin{align*}
\text{#} & \quad 3 & \quad 3 & \quad 2 & \quad 2 & \quad 3 & \quad 2 \\
\end{align*}
\]

\[
S(2.5) = \frac{15}{6} = 2 \frac{1}{2}
\]

Note for random BST \[
\frac{13}{5} = 2 \frac{3}{5}
\]
Keys \{ i_1, \ldots, i_n \} in Treep

\[ C(i, m) = \text{Event } \left[ i \text{ is compared to } m \text{ in Search}(m) \right] \]

Claim \[ \text{Prob}[C(i, m, 5)] = \frac{1}{m-i+1} \text{ if } i \leq m \]

\[ = \frac{1}{i-m} \text{ if } i > m \]

Treep construction at current game

![Diagram of Treep construction]

Informal argument: WLOG based in

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

\[ B_j = \text{Event} \left[ \text{\textit{it} \text{\textit{th}} \text{\textit{ch}\textit{d}t} \text{\textit{i}st} \text{\textit{f}\textit{\textit{r}}\textit{\textit{t}}} \text{\textit{t}\textit{\textit{h}}} \text{\textit{\textit{d}\textit{t}}} \text{\textit{\textit{\textit{\textit{i}d}n} in} [i_s, \ldots, m] \right] \]

\[ B_k \cap B_j = \emptyset \quad \text{for} \quad k \neq j \quad \text{and} \quad \Pr \left( \bigcap_{j=1}^{\infty} B_j \right) = 1 \quad (\times) \]

\[ \text{Conditional Probability: Let} \ A, B \ \text{be events} \]

\[ \Pr \left[ A \mid B \right] = \frac{\Pr \left[ A \cup B \right]}{\Pr \left[ B \right]} \]

\[ \sum_{i=1}^{m} \Pr \left[ C(i, m, i) \right] = \sum_{j=1}^{\infty} \Pr \left[ B_j \right] \Pr \left[ A \mid B_j \right] \quad \text{true by (}\times) \]

\[ \text{note} \quad \Pr \left[ A \mid B_j \right] = \frac{1}{m-i+1} \]

\[ \Pr \left[ A \right] = \sum_{i=1}^{\infty} \Pr \left[ B_j \right] \left( \frac{1}{m-i+1} \right) = \frac{1}{m-i+1} \sum_{j=1}^{\infty} \Pr \left[ B_j \right] = \frac{1}{m-i+1} \]
\[ S(m, s) = \# \text{ comparison search } m, s \]

\[ = \sum_{i \neq m} C(i, m, s) \]

\[ E(S(m, s)) = \sum E(C(i, m, s)) = \left\{ \text{Prob}[C(i, m, s)] \right\} \]

\[ = \sum_{i \leq m} \frac{1}{m-i+1} + \sum_{i > m} \frac{1}{i-m} \leq 2 \sum_{i=1}^{\frac{n}{2}} \frac{1}{i} = 2 \ln n \]

\[ \leq 2 \ln n + o(1) \]
Counting Rotations

2 cases: insert, delete

Delete: Move-to-Left
Claim: Expected # of rotations < 2
Claim: Possible pivots with x

1) Right-most nodes in left subtree of x
2) Left" "Right"

D_i = Event \{ i \text{ is a right-most node in } \text{ left subtree of } m \}

Don't Game Informal \Pr [D_i]

Pr [D_i] = (\frac{1}{m-i}) (\frac{1}{m-i})
\[ R_m = \text{# rotations move-to-leaf of } m \]

\[ E(R_m) \leq \sum_{i \neq m} E(D_i) \]

\[ \leq \sum_{i=0}^{m-1} \frac{1}{(m-i)(m-i)} + \sum_{i=m}^{n-m} \frac{1}{(i-m)(i-m)} \]

\[ \leq \sum_{i=0}^{m-1} \frac{1}{(i+1)i} + \sum_{i=1}^{n-m} \frac{1}{(i+1)i} \]

\[ \left( \frac{1}{(i+1)i} = \frac{1}{i} - \frac{1}{i+1} \right) \]

\[ \sum_{i=0}^{m-1} \frac{1}{i} - \sum_{i=1}^{n-m} \frac{1}{i+2} \]

\[ = \left( \frac{1}{0} - \frac{1}{m+1} \right) \]

\[ \leq 1 \]

\[ \leq 2 \]