Finding the rank of an element in a set

Use array:

Use array:

- Preprocess (sort)

- Partition

- rank(F) = ?

\( \Theta(n) \)

- OK if done once.
- Not for multiple queries

\( O(n\log n) \)

- Now all queries: \( O(1) \)

What if we want to insert/delete? \( \rightarrow \) bad \( O(n) \)
Finding the rank of an element in a Dynamic Set with Preprocessing

not dynamic ← sorting an array

RB-tree contains sorted letters
Now we can quickly restore sorted order
Store ranks... → bad (too many ranks change w/ insert)

Dynamic ?

Store subtree sizes
Using the augmented tree to find ranks

**Rank(H)**

Walk up from node, adding sizes of subtrees representing smaller #s

\[
\text{size}(l_H) = 0, \text{ walk up to } F \\
F \text{ is right child of } F, \text{ so count } F. \\
\text{size}(l_F) = 1 \quad \ldots \quad \text{sum} = 1 + 1 \\
\text{walk up to } C, \text{ count it.} \\
\text{size}(l_C) = 1 \quad \ldots \quad \text{increment sum by 1} \\
\text{walk up to } M, \text{ don't count it.} \\
\text{TOTAL} = 5 \quad (4 + 1 \text{ for } H)
\]
The balanced BST can be built in $\Theta(n\log n)$ time

Compute subtree sizes after building by postorder walk...

... or update path $\uparrow$
when inserting $\downarrow$

BUT...

we will need to rebalance
Can we update subtree sizes when inserting/deleting data?

Use a RB tree

when are subtree sizes affected? Rotations
AUGMENTED TREE TO FIND RANKS

- easy to find rank:
  - look at ancestor path & some adjacent subtree sizes

- subtree sizes can be updated when inserting and rebalancing

$O(\log n)$ per search/insertion/deletion
DYNAMIC SELECTION
find the i-th smallest element in a set

Static: $\Theta(n)$

Dynamic: $O(n \log n)$ preprocessing $\rightarrow$ balanced BST w/ subtree sizes

$O(\log n)$ query / insert / delete

(similar ... just need to see how to query)
Select\( (x, i) \) \hspace{1em} \text{get i-th element in subtree rooted at } x.
\[
k \leftarrow 1 + \text{size}(l_x) \hspace{1em} \text{\(l_x\): left child of } x
\]
if \( i = k \), return \( x \).
else if \( i < k \), return \( \text{Select}(l_x, i) \)
else \( i > k \), return \( \text{Select}(r_x, i-k) \)

Example: \( i = 5 \)

\[
\text{Select}(\text{root}, 5)
\]
\[
k \leftarrow 1 + 5
\]
\[
i < k \Rightarrow \text{Select}(c, 5)
\]
\[
k \leftarrow 1 + 1
\]
\[
i > k \Rightarrow \text{Select}(f, 3)
\]
\[
k \leftarrow 1 + 1
\]
\[
i > k \Rightarrow \text{Select}(h, 1)
\]
\[
k \leftarrow 1 + 0
\]
\[
i = k \Rightarrow \text{return } h
\]