Applications of amortized analysis
Binary search trees of bounded balance $\alpha \rightarrow \text{BB}(\alpha)$
(weight-balanced trees)
Binary search trees of bounded balance $\alpha \rightarrow \text{BB}(\alpha)$

(weight-balanced trees)

node with subtree weight $W$ and

left subtree weight $W_L$
right subtree weight $W_R$
Binary search trees of bounded balance $\alpha \rightarrow \text{BB}(\alpha)$ (weight-balanced trees)

For every node with subtree weight $W$ and

\[ W_L \geq \alpha W \]
\[ W_R \geq \alpha W \]

\[ 0 < \alpha \leq \frac{1}{2} \]
Binary search trees of bounded balance $\alpha$ \rightarrow BB(\alpha) (weight-balanced trees)

For every node with subtree weight $W$ and left subtree weight $W_L$ and right subtree weight $W_R$

\[ W_L \geq \alpha W \quad 0 < \alpha \leq \frac{1}{2} \]

\[ W_R \geq \alpha W \]

Height: ?
Binary search trees of bounded balance $\alpha \rightarrow \text{BB}(\alpha)$

(weight-balanced trees)

For every node with subtree weight $W$ and left subtree weight $W_L$ and right subtree weight $W_R$

$W_L \geq \alpha W$
$W_R \geq \alpha W$

$0 < \alpha \leq \frac{1}{2}$

Height: $H(n) \leq 1 + H((1-\alpha)n)$

\[ \leq \frac{1}{2} \]
Binary search trees of bounded balance $\alpha \rightarrow \text{BB}(\alpha)$ (weight-balanced trees)

For every node with subtree weight $W$ and left subtree weight $W_L$ and right subtree weight $W_R$

\[
W_L \geq \alpha W \\
W_R \geq \alpha W
\]

$0 < \alpha \leq \frac{1}{2}$

Height: $H(n) \leq 1 + H((1-\alpha)n) \geq \frac{1}{2}$

standard geometric series:

$H \sim \log \frac{1}{(1-\alpha)n} = \Theta(\log n)$
Insertion/Deletion in $BB(\alpha)$ weight-balanced tree
Insertion/Deletion in BB(α) weight-balanced tree

Any ancestor could now have: \( W_R < \alpha W \) (wlog)
Insertion/Deletion in BB(α) weight-balanced tree

Any ancestor could now have:

\[ W_R < \alpha W \] (wlog)

Solution: rotate
Insertion/Deletion in BB(\(\alpha\)) weight-balanced tree

FYI
actually, 2 types of rotation
\(\downarrow\) standard & “split”

Any ancestor could now have:
\(WR < \alpha W\) (wlog)

Solution: rotate

... depends on amount of imbalance
Insertion/Deletion in $BB(\alpha)$ weight-balanced tree

Any ancestor could now have: $W_R < \alpha W$ (wlog)

Solution: rotate

FYI

- Actually, 2 types of rotation:
  - Standard & “split”

... depends on amount of imbalance

Works for $\alpha < 1 - \frac{1}{\sqrt{2}} \approx 0.3$ // Proof involves some annoying counting

Can walk up & rebalance ancestors: $O(\log n)$
Insertion/Deletion in BB(α) weight-balanced tree

\[ \alpha < 1 - \frac{1}{\sqrt{2}} \]

FYI
Insertion/Deletion in BB(α) weight-balanced tree

lower bound: \( \frac{2}{11} < \alpha \)  
\( \alpha < 1 - \frac{1}{\sqrt{2}} \)

FYI
Insertion/Deletion in BB(α) weight-balanced tree

Lower bound:

\[ \frac{2}{11} < \alpha \quad \text{and} \quad \alpha < 1 - \frac{1}{\sqrt{2}} \]

Claim:

can rebalance for $\alpha$ outside this range with more complicated rotations

FYI
Insertion/Deletion in BB(α) weight-balanced tree

lower bound:
\[ \frac{2}{11} < \alpha < 1 - \frac{1}{\sqrt{2}} \]

Claim:
can rebalance for \( \alpha \) outside this range with more complicated rotations

Instead, we will avoid rotations
α-violation may occur for any ancestor

Let v be highest violation
$\alpha$-violation may occur for any ancestor

Let $v$ be highest violation
\( \alpha \)-violation may occur for any ancestor

Let \( v \) be highest violation

Rebuild \( \text{subtree}(v) \) cost?
α-violation may occur for any ancestor

Let v be highest violation

Rebuild subtree(v) \quad O(\log n) + \Theta(\text{size}(v)) = O(n)

Amortize ... ?
$\alpha$-violation may occur for any ancestor

Let $v$ be highest violation

Rebuild subtree($v$)  $O(\log n) + \Theta(\text{size}(v)) = O(n)$

$$\Phi = \sum_{\text{all nodes}} \frac{1}{1-2\alpha} |W_L-W_R| \quad \rightarrow \quad \text{technically, only if } \text{diff} \geq 2$$

$\text{(diff} = 1 \text{ unavoidable)}$

e.g. $\alpha = \frac{1}{3}$ : $\Phi = 3 \cdot \sum |W_L-W_R|$
$\alpha$-violation may occur for any ancestor

Let $v$ be highest violation

Rebuild subtree($v$) \[ O(\log n) + \Theta(\text{size}(v)) = O(n) \]

$$\Phi = \sum_{\text{all nodes}} \frac{1}{1 - 2\alpha} |W_L - W_R| \rightarrow \text{technically, only if } \text{diff} \geq 2$$

$$(\text{diff} = 1 \text{ unavoidable})$$

e.g. $\alpha = \frac{1}{3}$ : $\Phi = 3 \cdot \sum |W_L - W_R|$

Regular insert $\rightarrow$ $\Delta \Phi = 3 \cdot \sum \Delta |W_L - W_R|$
\( \alpha \)-violation may occur for any ancestor

Let \( v \) be highest violation

Rebuild subtree(\( v \)) \( O(\log n) + \Theta(\text{size}(v)) = O(n) \)

\[
\Phi = \sum_{\text{all nodes}} \frac{1}{1-2\alpha} |W_L-W_R| \quad \rightarrow \text{technically, only if } \text{diff} \geq 2 \quad (\text{diff} = 1 \text{ unavoidable})
\]

e.g. \( \alpha = \frac{1}{3} \) : \( \Phi = 3 \cdot \sum |W_L-W_R| \)

Regular insert \( \rightarrow \Delta\Phi = 3 \cdot \sum_{\text{all ancestors}} \Delta|W_L-W_R| \leq 3\log n \)
\( \alpha \)-violation may occur for any ancestor

Let \( v \) be highest violation

Rebuild subtree(\( v \))  \( O(\log n) + \Theta(\text{size}(v)) = O(n) \)

\[
\Phi = \sum_{\text{all nodes}} \frac{1}{1-2\alpha} |W_L - W_R|
\]

\( \rightarrow \) technically, only if \( \text{diff} \geq 2 \)

(diff = 1 unavoidable)

e.g. \( \alpha = \frac{1}{3} \) : \( \Phi = 3 \cdot \sum |W_L - W_R| \)

Regular insert \( \rightarrow \) \( \Delta \Phi = 3 \cdot \sum_{\text{all ancestors}} \Delta |W_L - W_R| \leq 3 \log n \)

\( \hat{C} \leq 4 \cdot \log n \)
$\alpha$-violation may occur for any ancestor

Let $v$ be highest violation

Rebuild subtree($v$)  $O(\log n) + \Theta(\text{size}(v)) = O(n)$

$$\Phi = \sum_{\text{all nodes}} \frac{1}{1-2\alpha} |W_L - W_R| \quad \rightarrow \text{technically, only if } \text{diff} \geq 2$$

(diff = 1 unavoidable)

E.g. $\alpha = \frac{1}{3}$: $\Phi = 3 \cdot \sum |W_L - W_R|$

Rebuild($v$) $\rightarrow \Delta \Phi = 3 \cdot \sum_{\text{all } x \in \text{tree}(v)} \Delta |W_L - W_R|$
\( \alpha \)-violation may occur for any ancestor

Let \( v \) be highest violation

Rebuild subtree(\( v \)) \quad O(\log n) + \Theta(\text{size}(v)) = O(n)

\[
\Phi = \sum_{\text{all nodes}} \frac{1}{1-2\alpha} |W_L - W_R|
\rightarrow \text{technically, only if diff} \geq 2
\quad \text{(diff = 1 unavoidable)}
\]

\( \alpha = \frac{1}{3} \): \quad \Phi = 3 \cdot \sum |W_L - W_R|

Rebuild(\( v \)) \rightarrow \Delta \Phi = 3 \cdot \sum_{\text{all } x \in \text{tree}(v)} \Delta |W_L - W_R|

\( \forall \) every \( \Delta \Phi \) term \leq 0
\( \alpha \)-violation may occur for any ancestor

Let \( v \) be highest violation

Rebuild subtree(\( v \)) \hspace{1cm} O(\log n) + \Theta(\text{size}(v)) = O(n)

\[
\Phi = \sum_{\text{all nodes}} \frac{1}{1-2\alpha} |W_L - W_R|
\]

\( \Phi \rightarrow \) technically, only if diff \( \geq 2 \)

\( \text{diff} = 1 \text{ unavoidable} \)

\[
eq \frac{1}{3} : \hspace{1cm} \Phi = 3 \cdot \sum |W_L - W_R|
\]

Rebuild(\( v \)) \( \rightarrow \Delta \Phi = 3 \cdot \sum_{\text{all } x \in \text{tree}(v)} \Delta |W_L - W_R| \leq 3 \cdot \Delta |W_L^v - W_R^v|
\]

\( \geq \) every \( \Delta \Phi \) term \( \leq 0 \)
\( \alpha \)-violation may occur for any ancestor

Let \( v \) be highest violation

Rebuild subtree(\( v \)) \( O(\log n) + \Theta(\text{size}(v)) = O(n) \)

\[
\Phi = \sum_{\text{all nodes}} \frac{1}{1 - 2\alpha} |W_L - W_R| \quad \rightarrow \text{technically, only if diff } \geq 2
\]

(diff = 1 unavoidable)

\[ e.g. \; \alpha = \frac{1}{3} : \; \Phi = 3 \sum |W_L - W_R| \]

\[ \text{Rebuild}(v) \rightarrow \Delta \Phi = 3 \cdot \sum_{\text{all } x \in \text{intree}(v)} |W_L - W_R| \leq 3 \cdot |W_L^v - W_R^v| \]

Every \( \Delta \Phi \) term \( \leq 0 \)

violation: \( \text{wlog } W_R < \frac{1}{3} W \)
$\alpha$-violation may occur for any ancestor

Let $v$ be highest violation

$\text{Rebuild subtree}(v) \quad O(\log n) + \Theta(\text{size}(v)) = O(n)$

$$\Phi = \sum_{\text{all nodes}} \frac{1}{1-2\alpha} |W_L - W_R|$$

$\rightarrow$ technically, only if $\text{diff} \geq 2$

(diff = 1 unavoidable)

Example: $\alpha = \frac{1}{3} \Rightarrow \Phi = 3 \cdot \sum |W_L - W_R|$

Rebuild($v$) $\rightarrow$ $\Delta \Phi = 3 \cdot \sum_{\text{all x in tree(v)}} \Delta |W_L - W_R| \leq 3 \cdot \Delta |W_L^v - W_R^v|$

Every $\Delta \Phi$ term $\leq 0$

Violation: wlog $W_R < \frac{1}{3}W \quad \frac{2}{3}W < W_L$
$\alpha$-violation may occur for any ancestor

Let $v$ be highest violation

$\text{Rebuild subtree}(v) \quad O(\log n) + \Theta(\text{size}(v)) = O(n)$

$$\Phi = \sum_{\text{all nodes}} \frac{1}{1 - 2\alpha} |W_L - W_R| \quad \rightarrow \quad \text{technically, only if diff} \geq 2$$

(diff = 1 unavoidable)

e.g. $\alpha = \frac{1}{3}: \quad \Phi = 3 \cdot \sum |W_L - W_R|$

Rebuild$(v) \rightarrow \Delta \Phi = 3 \cdot \sum_{\text{all } x \in \text{tree}(v)} \Delta |W_L - W_R| \leq 3 \cdot \Delta |W_L^v - W_R^v|$

Every $\Delta \Phi$ term $\leq 0$

violation: $\omega\log W_R < \frac{1}{3} W < \frac{2}{3} W < W_L$
\(\alpha\)-violation may occur for any ancestor

Let \(v\) be highest violation

Rebuild subtree(\(v\)) \(O(\log n) + \Theta(\text{size}(v)) = O(n)\)

\[
\Phi = \sum_{\text{all nodes}} \frac{1}{1-2\alpha} |W_L - W_R|
\]

\(\rightarrow\) technically, only if diff \(\geq 2\)

(diff = 1 unavoidable)

e.g. \(\alpha = \frac{1}{3}\) : \(\Phi = 3 \cdot \sum |W_L - W_R|\)

Rebuild(\(v\)) \(\rightarrow\) \(\Delta \Phi = 3 \cdot \sum \Delta |W_L - W_R| \leq 3 \cdot \Delta |W_L^v - W_R^v|\)

\(\forall\) every \(\Delta \Phi\) term \(\leq 0\)

\(\Rightarrow \) Rebuild(\(v\)) \(\rightarrow\) \(\Delta \Phi \leq 3 \cdot [O - \frac{1}{3}W] \approx -\text{size}(v)\)

\(\Phi_i - \Phi_{i-1}\)

\(\text{violation: wlog } W_R < \frac{1}{3}W < \frac{2}{3}W < W_L\)
α-violation may occur for any ancestor

Let v be highest violation

Rebuild subtree(v) \( O(\log n) + \Theta(\text{size}(v)) = O(n) \)

\[ \Phi = \sum_{\text{all nodes}} \frac{1}{1-2\alpha} |W_L - W_R| \quad \rightarrow \text{technically, only if diff } \geq 2 \]

(diff = 1 unavoidable)

\[ \Delta \Phi = \frac{1}{1-2\alpha} \cdot \sum_{\text{all x in tree}(v)} \Delta |W_L - W_R| \leq \frac{1}{1-2\alpha} \cdot \Delta |W_L - W_R| \]

\( \text{every } \Delta \Phi \text{ term } \leq 0 \)

\( W_R < \alpha W < (1-\alpha) W < W_L \)

\[ \Phi_i - \Phi_{i-1} \leq \frac{1}{1-2\alpha} \cdot [O - (1-2\alpha)W] \sim \text{size}(v) \]
An application of BB(\(\alpha\))

Dynamic high-dimensional multilayered range trees

\(\leftarrow\) not easy to update efficiently with rotations

FYI
1D range counting

Store size of each subtree

$k = 6$
Every X-range is represented by $O(\log n)$ nodes.

For each node, create a new (aux.) tree containing all nodes of subtree, sorted by Y.

2D range counting
3D: tree of trees of trees
More results that use amortized analysis
SCAPEGOAT TREES: amortized balanced dynamic BST

$n = \# \text{ keys}$

$q = \text{ variable s.t. } 9/2 \leq n \leq q$

$\alpha = \text{ balance factor, } \frac{1}{2} < \alpha < 1$

determines \underline{max allowed height}.

$h \leq \log_{\frac{1}{\alpha}} q$

$\leq \log_{\frac{1}{\alpha}} 2n = \log_{\frac{1}{\alpha}} n + O(1)$
Heap building: we have seen $O(n)$ instead of $O(n \log n)$
Heap building: we have seen $O(n)$ instead of $O(n \log n)$

Binomial heaps: collections of binomial trees
Heap building: we have seen $O(n)$ instead of $O(n \log n)$

Binomial heaps: collections of binomial trees

- size = power of 2
- only one of each size
- trivially mergeable
Heap building: we have seen $O(n)$ instead of $O(n\log n)$

Binomial heaps: collections of binomial trees

- size = power of 2
- only one of each size
- trivially mergeable

merging two b.heaps: like binary addition ... $O(\log n)$
Heap building: we have seen $O(n)$ instead of $O(n \log n)$

Binomial heaps: collections of binomial trees

- size = power of 2
- only one of each size
- trivially mergeable

merging two b.heaps: like binary addition ... $O(\log n)$

insert into b.heap: $O(\log n)$ worst case
Heap building: we have seen $O(n)$ instead of $O(n\log n)$

Binomial heaps: collections of binomial trees

- size = power of 2
- only one of each size
- trivially mergeable

Merging two b.heaps: like binary addition ... $O(\log n)$

Insert into b.heap: $O(\log n)$ worst case

But $O(1)$ amortized
<table>
<thead>
<tr>
<th>Heaps:</th>
<th>Regular</th>
<th>Binomial</th>
<th>Fibonacci</th>
</tr>
</thead>
<tbody>
<tr>
<td>Report Min:</td>
<td>$O(1)$</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>Extract Min:</td>
<td>$O(\log n)$</td>
<td>$O(\log n)$ (amortized)</td>
<td>$O(n)$ ($\log n$) amortized</td>
</tr>
<tr>
<td>Insert:</td>
<td>$O(\log n)$</td>
<td>$O(1)$ amortized</td>
<td>$O(1)$</td>
</tr>
<tr>
<td>Decrease Key:</td>
<td>$O(\log n)$</td>
<td>--</td>
<td>$O(1)$ amortized</td>
</tr>
<tr>
<td>Delete:</td>
<td>$O(\log n)$</td>
<td>--</td>
<td>$O(n)$ amortized</td>
</tr>
<tr>
<td>Merge/Union:</td>
<td>$O(n)$</td>
<td>$O(\log n)$</td>
<td>$O(1)$ (£)</td>
</tr>
<tr>
<td>HEAPS:</td>
<td>REGULAR</td>
<td>BINOIMAL</td>
<td>QUAKE</td>
</tr>
<tr>
<td>---------------</td>
<td>---------</td>
<td>----------</td>
<td>----------------</td>
</tr>
<tr>
<td>REPORT MIN:</td>
<td>$O(1)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>EXTRACT MIN:</td>
<td>$O(\log n)$</td>
<td></td>
<td>$O(n)$</td>
</tr>
<tr>
<td>INSERT:</td>
<td>$O(\log n)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$O(1)$ amortized</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DECREASE KEY:</td>
<td>$O(\log n)$</td>
<td></td>
<td>$O(1)$</td>
</tr>
<tr>
<td>DELETE:</td>
<td>$O(\log n)$</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$O(n)$ amortized</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MERGE/UNION:</td>
<td>$O(n)$</td>
<td>$O(\log n)$</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$O(1)$</td>
</tr>
</tbody>
</table>
Splay trees: whenever you access a node, bring it to the top

Amortized cost $O(\log n)$

\[
\text{\texttt{(if no } G \text{ then } \xrightarrow{P} \xrightarrow{P})}
\]