Red-Black Trees

Definition
Analysis
Clusters (not in book)
Operations
Red-black trees are binary search trees that satisfy:

1. Every node is either red or black.
2. The root is black.
3. If a node is red, then its parent is black.
4. For a given node, every path to a NIL has the same number of black nodes, called black-height.

The height of a red-black tree is balanced, that is, its height is $\Theta(\lg n)$.

Specifically, a red-black tree with $n$ nodes has height $h \leq 2\lg(n + 1)$. 
Let $h$ be the height and $b$ be the black-height.

Point 1: $h \leq 2b$ for all red-black trees.

Basis: A r-b tree with black-height 1 has height 0 or height 1.

Assume: Assume r-b trees with black-height $b$ have height $\leq 2b$.

Show: r-b trees with black-height $b + 1$ have height $\leq 2b + 2$

Induction: Increasing the black-height by 1 increases the height by at most 2.
Balanced Proof Part 2

Point 2: Black-height $b$ implies $n \geq 2^b - 1$

Basis: If $b = 1$, then $n \geq 1 = 2^1 - 1$

Assume: If $b = k - 1$, then $n \geq 2^{k-1} - 1$

Show: If $b = k$, then $n \geq 2^k - 1$

Induction: 2 subtrees with black-height $k - 1$ implies $n \geq 1 + 2(2^{k-1} - 1) = 2^k - 1$

Combining the two points:
$n \geq 2^b - 1 \geq 2^{h/2} - 1$ implies $h \leq 2 \lg(n + 1)$
(not in book) To explain the operations, define a cluster to be a black node and its red children.

- A cluster has one, two, or three nodes.
- One node in a cluster is black.
- Any other nodes are red and are children of the black node.
- A cluster must satisfy the binary search and red-black tree properties.
Clusters Part 2

red-black trees
proof
clusters
▷ clusters 2
insert
delete
delete 2
**RB-Insert**\((T, x)\)

Tree-Insert\((T, x)\) into a “leaf” cluster

\[
\text{cluster} \leftarrow \text{cluster containing } x
\]

while \text{cluster} has too many nodes

“split” \text{cluster}

\[
\text{cluster} \leftarrow \text{parent of } \text{cluster}
\]
**RB-DELETE**\((T, z)\)

**Tree-Delete**\((T, z)\)

- Initialize the cluster to be the leaf cluster with one less node.
- While the cluster is not empty:
  - “Splice” or “shift” into the cluster.
  - Update the cluster to be the parent of the current cluster.

**Diagram:**

```
.... A ....
  \  /    \\
 /  \  \    \\
∅   B   T2 T3
  \  /    \\
 T1       
```

**SPLICE**

```
... ... ...
    \    /  \\
    /  \  \\
 A   B  T1 T2 T3
```

```
       ... ...
             /  \\
             /  \  \\
         A   B  T1 T2 T3
```
Delete Part 2

\[ \text{\ldots A \ldots} \]
\[ \varnothing \quad B \quad C \]
\[ \text{T}_1 \quad \text{T}_2 \quad \text{T}_3 \quad \text{T}_4 \]

\[ \text{\ldots B \ldots} \]
\[ \text{A} \quad \text{C} \]
\[ \text{T}_1 \quad \text{T}_2 \quad \text{T}_3 \quad \text{T}_4 \]

\[ \text{\ldots A \ldots} \]
\[ \varnothing \quad B \quad C \quad D \]
\[ \text{T}_1 \quad \text{T}_2 \quad \text{T}_3 \quad \text{T}_4 \quad \text{T}_5 \]

\[ \text{\ldots B \ldots} \]
\[ \text{A} \quad \text{C} \quad \text{D} \]
\[ \text{T}_1 \quad \text{T}_2 \quad \text{T}_3 \quad \text{T}_4 \quad \text{T}_5 \]