MCS-375: Algorithms: Analysis and Design Handout #DS5
San Skulrattanakulchai
Gustavus Adolphus College September 22, 2014

Red-black Trees

- A red-black tree (rbt) is a bst that is full, and each node is either colored red or black, and only the internal nodes are key-bearing. Furthermore, it satisfies the following three properties.

  1. (Black Root Property) The root is black.
  2. (Black Leaves Property) Every leaf (called NIL) is black.
  3. (Red Isolation Property) No red node has a red child.
  4. (Black Height Property) Every simple path from the root to a leaf contains the same number of black nodes.

[EXAMPLE RBT PICTURE HERE]

Notes

0. When an RBT is drawn, the NIL nodes are usually not drawn even though they are there. This avoids clutter.

1. The NIL nodes are real nodes (not empty). They act like sentinels, thus help simplify the code.

2. Red Isolation Property can be stated as "No red node has a red parent," or "No two red nodes are joined by an edge."

3. Black Height Property implies that *for any fixed node x, every simple path from x to a leaf contains the same number of black nodes.*

4. In addtion to *key*, p, $left$, and *right* fields, each node in an rbt has a *color* field.

5. In addition to ordinary height, every rbt also has a *black height*.

First define the black height of any node x to be the number of black nodes on any simple path from x to a NIL node, not counting x itself.

Now define the black height of a tree to be the black height of its root.

Note that NIL nodes contribute to both black height and ordinary height.

6. The black root property is in fact not essential. The reason for having it is similar to having NIL nodes as real nodes instead of being empty (NULL). Both help simplify the code by avoiding complicating trivial cases.

- The following theorem shows rbt's are "balanced."

Theorem (Red-Black Tree Height Theorem)**.** *The height of an rbt T with n internal nodes is $\leq 2\log(n+1)$.*

Proof. Let b (h) be the black (ordinary) height of T. By definition of black height, we know that for each $i = 0, 1, 2, \ldots, b-1$, level i is filled, i.e., it has the greatest possible number of nodes, i.e., 2^i nodes. We thus have

$$n \geq 1 + 2 + 2^2 + \cdots + 2^{b-1} = 2^b - 1.$$

Consider a longest root-to-NIL simple path P in T. Let P have r red nodes. By the Black Root Property, the Red Isolation Property, and the Black Leaves Property, we conclude that every red node has a black child; thus $r \leq b$. Therefore, $h = r + b \leq b + b = 2b$, i.e. $b \geq h/2$. Therefore, $n \geq 2^b - 1 \geq 2^{h/2} - 1$, ie, $2^{h/2} \leq n+1$. Taking logarithm of both sides then multiplying both sides by 2 gives $h \leq 2\log(n+1)$ as desired. $\square$