MCS-375: Algorithms: Analysis and Design Handout #DS7
San Skulrattanakulchai
Gustavus Adolphus College Sep 25, 2014

Deletion Algorithm for Red-Black Trees

CLRS: Ch 13.4

To delete a node z from a red-black tree:

Step 1. Delete z using the regular bst DELETE procedure.

Step 2. Fix up the tree to restore the red-black tree properties.

Step 1 will physically delete a node z. It also defines nodes y and x as follows.
Node y is defined in term of z. If z has ≤ 1 non-`NIL` child, y is z itself. If z has exactly 2 non-`NIL` children, y is the successor of z.
Node x is defined to be the node that will be put into y's original place. If y has exactly one non-`NIL` child, node x is that non-`NIL` child of y. If both of y's children are `NIL`, then a `NIL` is actually put into y's place, and x is that chosen `NIL` node. (Note that y has no more than 1 non-`NIL` child.)

Step 2 consists of 2 simple cases, and a loop containing the Pushup, Rearrange & Reduce Cases.

Simple Case 1. If y was red, then we are done!
Simple Case 2. If y had a red non-`NIL` child x, then blacken x and we are done!

From now on assume the 2 simple cases don't apply.
So y was black and both of its children were `NIL`.
At this point the Black Height Property is violated.
However, Black Height Property holds if we count x as 2 ("double") blacks.
Keep applying the appropriate case from below until done.

Pushup Case. If x's sibling and nephews are all black, push a black up 1 level.

PUSHUP PICTURE HERE

If the new x is red or is the root, then blacken it and we are done!

Rearrange Case. If x's sibling is black and at least one nephew is red, then rearrange the tree to make it valid and stop. We are done.

REARRANGE PICTURE HERE

Reduce Case. If x's sibling is red (x's parent and nephews must be black), then rotate $p[x]$ and recolor so that x's sibling is black and then follow the Pushup or Rearrange Case.

REDUCE PICTURE HERE

Timing

The deletion algorithm takes $O(\log n)$ time.

Step 1, the bst delete, takes $O(\log n)$ time.

The total time spent in the Pushup Case is $O(\log n)$.

The total time spent in the Rearrange and Reduce Cases is $O(1)$.

Notes

1. Pushup Case corresponds to Case 2 of CLRS. Rearrange Case corresponds to Case 3–4 of CLRS. Reduce Case corresponds to Case 1 of CLRS.

Exercises

1. Explain why the Reduce Case uses total time $O(1)$, i.e., show that after the Reduce Case is executed we execute either the Pushup or Rearrange Case, stopping in both.

2. Show that the deletion routine does at most 3 rotations.