Red Black Trees

Top-Down Deletion

Recall the rules for BST deletion

1. If vertex to be deleted is a leaf, just delete it.
2. If vertex to be deleted has just one child, replace it with that child
3. If vertex to be deleted has two children, replace the **value** of by it's in-order predecessor's value then delete the in-order predecessor (a recursive step)

What can go wrong?

1. If the delete node is red?

   Not a problem – no RB properties violated

2. If the deleted node is black?

   If the node is not the root, deleting it will change the black-height along some path

The goal of T-D Deletion

- To delete a red leaf
- How do we ensure that's what happens?
  - As we traverse the tree looking for the leaf to delete, we change every node we encounter to red.
  - If this causes a violation of the RB properties, we fix it

Bottom-Up vs. Top-Down

- Bottom-Up is recursive
  - BST deletion going down the tree (winding up the recursion)
  - Fixing the RB properties coming back up the tree (unwinding the recursion)
- Top-Down is iterative
  - Restructure the tree on the way down so we don't have to go back up

Terminology

- Matching Weiss text section 12.2
  - X is the node being examined
  - T is X's sibling
  - P is X's (and T's) parent
  - R is T's right child
  - L is T's left child
- This discussion assumes X is the left child of P. As usual, there are left-right symmetric cases.

Basic Strategy

- As we traverse the tree, we change every node we visit, X, to Red.
- When we change X to Red, we know
  - P is also Red (we just came from there)
  - T is black (since P is Red, it's children are Black)

Step 1 – Examine the root

1. If both of the root's children are Black
   a. Make the root Red
   b. Move X to the appropriate child of the root
   c. Proceed to step 2
2. Otherwise designate the root as X and proceed to step 2B.

Step 2 – the main case

As we traverse down the tree, we continually encounter this situation until we reach the node to be deleted

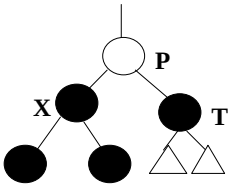
X is Black, P is Red, T is Black

We are going to color X Red, then recolor other nodes and possibly do rotation(s) based on the color of X's and T's children

2A. X has 2 Black children

2B. X has at least one Red child

Case 2A
X has two Black Children



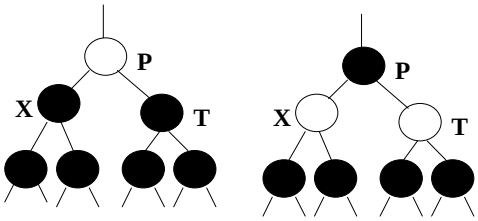
2A1. T has 2 Black Children

2A2. T's left child is Red

2A3. T's right child is Red

** if both of T's children are Red, we can do either 2A2 or 2A3

Case 2A1
X and T have 2 Black Children

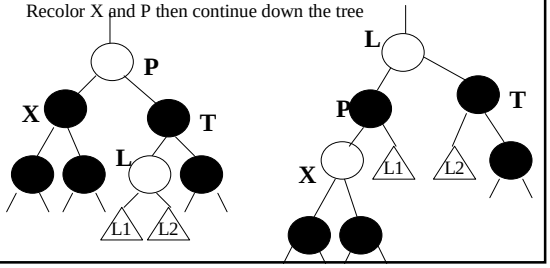


Just recolor X, P and T and move down the tree

Case 2A2

X has 2 Black Children and T's Left Child is Red

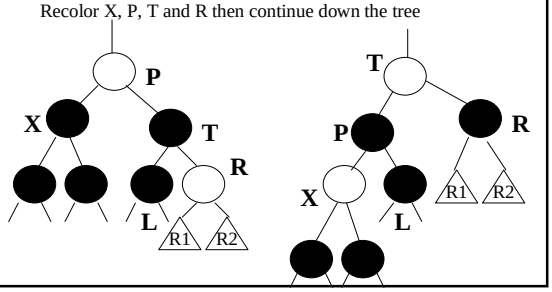
Rotate L around T, then L around P
Recolor X and P then continue down the tree



Case 2A3

X has 2 Black Children and T's Right Child is Red

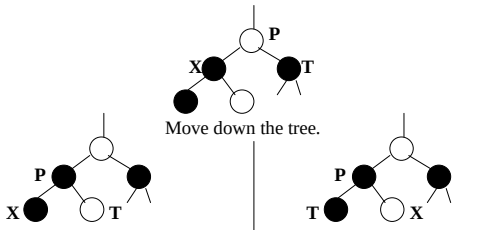
Rotate T around P
Recolor X, P, T and R then continue down the tree



Case 2B
X has at least one Red child

Continue down the tree to the next level
If the new X is Red, continue down again
If the new X is Black (T is Red, P is Black)
- Rotate T around P
- Recolor P and T
- Back to main case – step 2

Case 2B Diagram



If move to Black child (2B2)
Rotate T around P; Recolor P and T
Back to step 2, the main case

If move to the Red child (2B1)
Move down again

Step 3

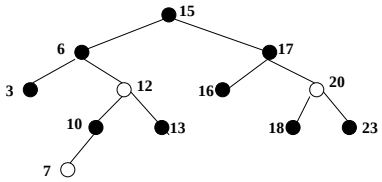
Eventually, find the node to be deleted – a leaf or a node with one non-null child that is a leaf.

Delete the appropriate node as a Red leaf

Step 4

Color the Root Black

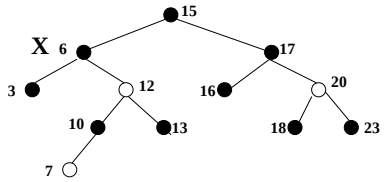
Example 1
Delete 10 from this RB Tree



Step 1 – Root has 2 Black children. Color Root Red

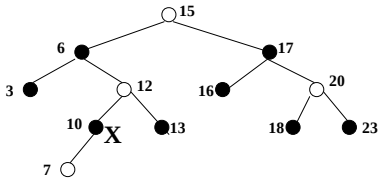
Descend the tree, moving X to 6

Example 1 (cont'd)



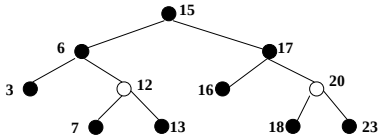
One of X's children is Red (case 2B). Descend down the tree, arriving at 12. Since the new X (12) is also Red (2B1), continue down the tree, arriving at 10.

Example 1 (cont'd)



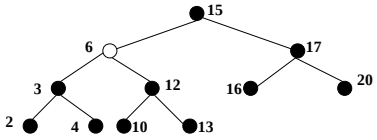
Step 3 -Since 10 is the node to be deleted, replace it's value with the value of it's only child (7) and delete 7's red node

Example 1 (cont'd)



The final tree after 7 has replaced 10 and 7's red node deleted and (step 4) the root has been colored Black.

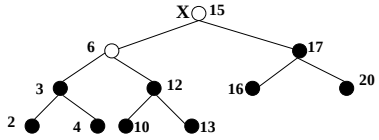
Example 2
Delete 10 from this RB Tree



Step 1 – the root does not have 2 Black children.

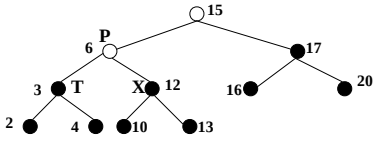
Color the root red, Set X = root and proceed to step 2

Example 2 (cont'd)



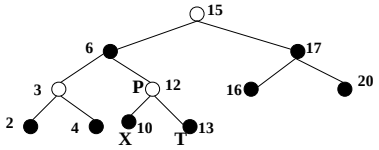
X has at least one Red child (case 2B). Proceed down the tree, arriving at 6. Since 6 is also Red (case 2B1), continue down the tree, arriving at 12.

Example 2 (cont'd)



X has 2 Black children. X's sibling (3) also has 2 black children. Case 2A1– recolor X, P, and T and continue down the tree, arriving at 10.

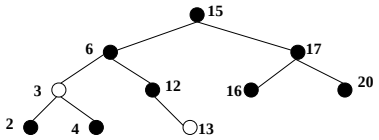
Example 2 (cont'd)



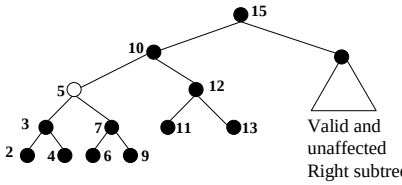
X is now the leaf to be deleted, but it's Black, so back to step 2.
X has 2 Black children and T has 2 Black children – case 2A1

Recolor X, P and T.
Step 3 -- Now delete 10 as a red leaf.
Step 4 -- Recolor the root black

Example 2 Solution



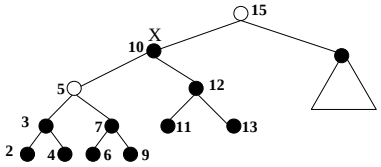
Example 3
Delete 11 from this RB Tree



Step 1 – root has 2 Black children. Color Root red.

Set X to appropriate child of root (10)

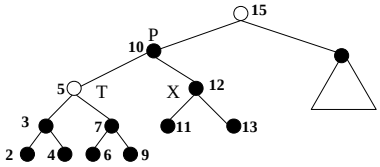
Example 3 (cont'd)



X has one Red child (case 2B)

Traverse down the tree, arriving at 12.

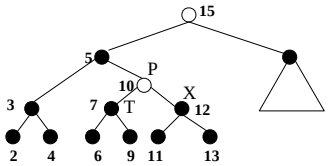
Example 3 (cont'd)



Since we arrived at a black node (case 2B2) assuring T is red and P is black), rotate T around P, recolor T and P

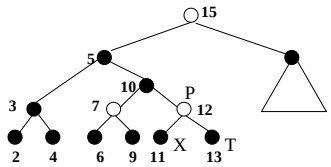
Back to step 2

Example 3 (cont'd)



Now X is Black with Red parent and Black sibling.
X and T both have 2 Black children (case 2A1)
Just recolor X, P and T and continue traversal

Example 3 (cont'd)



Having traversed down the tree, we arrive at 11, the leaf to be deleted, but it's Black, so back to step 2.
X and T both have two Black children. Recolor X, P and T.
Step 3 -- delete 11 as a red leaf.
Step 4 -- Recolor the root black

Example 3 Solution