Red-Black Tree | Set 2 (Insert)

In the previous post, we discussed introduction to Red-Black Trees. In this post, insertion is discussed.

3.5

In AVL tree insertion, we used rotation as a tool to do balancing after insertion caused imbalance. In Red-Black tree, we use two tools to do balancing.

1) Recoloring
2) Rotation

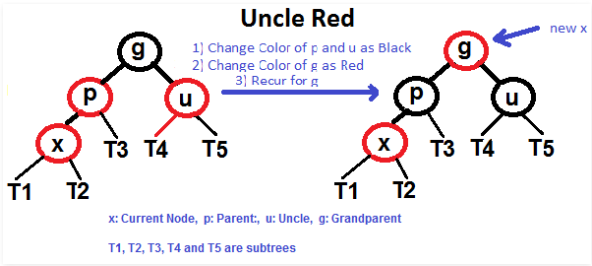
We try recoloring first, if recoloring doesn't work, then we go for rotation. Following is detailed algorithm. The algorithms has mainly two cases depending upon the color of uncle. If uncle is red, we do recoloring. If uncle is black, we do rotations and/or recoloring.

Color of a NULL node is considered as BLACK.

Let x be the newly inserted node.
1) Perform standard BST insertion and make the color of newly inserted nodes as RED.

2) If x is root, change color of x as BLACK (Black height of complete tree increases by 1).

3) Do following if color of x's parent is not BLACK or x is not root.
….**a) If x's uncle is RED** (Grand parent must have been black from property 4)
……..**(i)** Change color of parent and uncle as BLACK.
……..**(ii)** color of grand parent as RED.
……..**(iii)** Change x = x's grandparent, repeat steps 2 and 3 for new x.

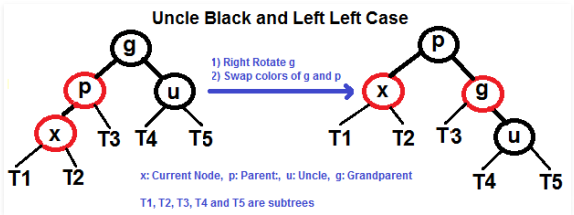


….**b) If x's uncle is BLACK**, then there can be four configurations for x, x's parent (**p**) and x's grandparent (**g**) (This is similar to AVL Tree)
……..**i)** Left Left Case (p is left child of g and x is left child of p)
……..**ii)** Left Right Case (p is left child of g and x is right child of p)
……..**iii)** Right Right Case (Mirror of case a)
……..**iv)** Right Left Case (Mirror of case c)

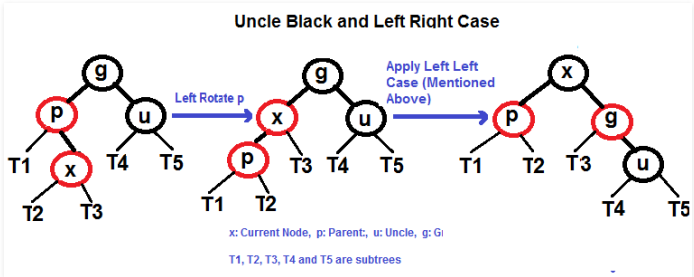
Following are operations to be performed in four subcases when uncle is BLACK.

All four cases when Uncle is BLACK

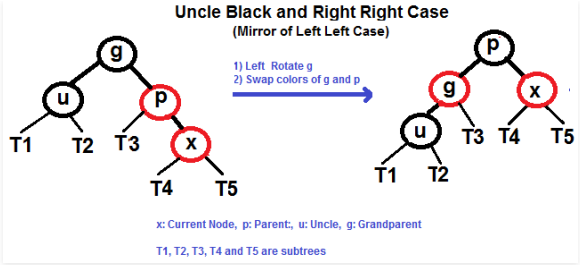
Left Left Case (See g, p and x)



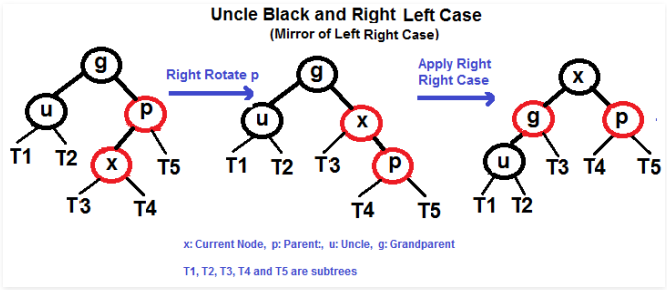
Left Right Case (See g, p and x)



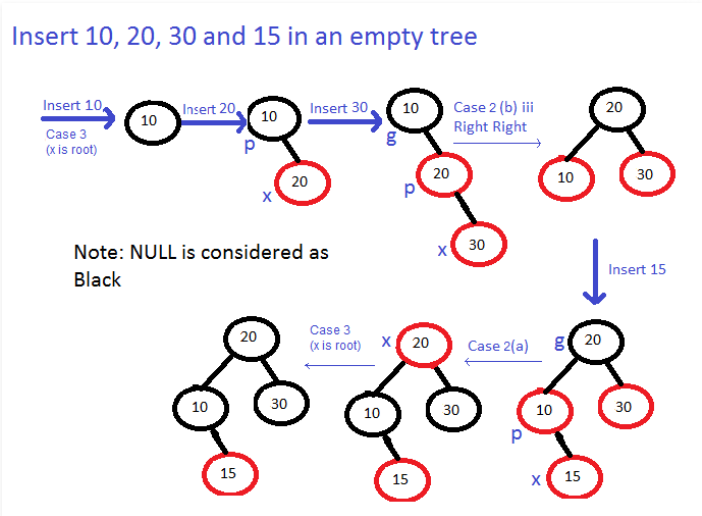
Right Right Case (See g, p and x)



Right Left Case (See g, p and x)

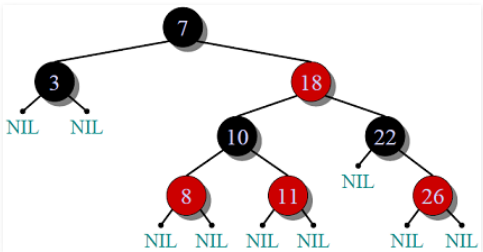


Examples of Insertion



Exercise:

Insert 2, 6 and 13 in below tree.



Please refer **C Program for Red Black Tree Insertion** for complete implementation of above algorithm.

Red-Black Tree | Set 3 (Delete)

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

GATE CS Corner Company Wise Coding Practice

Advanced Data Structure Self-Balancing-BST

Login to Improve this Article

Recommended Posts:

Red-Black Tree | Set 3 (Delete)
Red-Black Tree | Set 1 (Introduction)
AVL Tree | Set 1 (Insertion)
Interval Tree
AVL Tree | Set 2 (Deletion)
Sparse Table
Minimum Word Break
Count greater nodes in AVL tree
Burrows – Wheeler Data Transform Algorithm
Segment tree | Efficient implementation