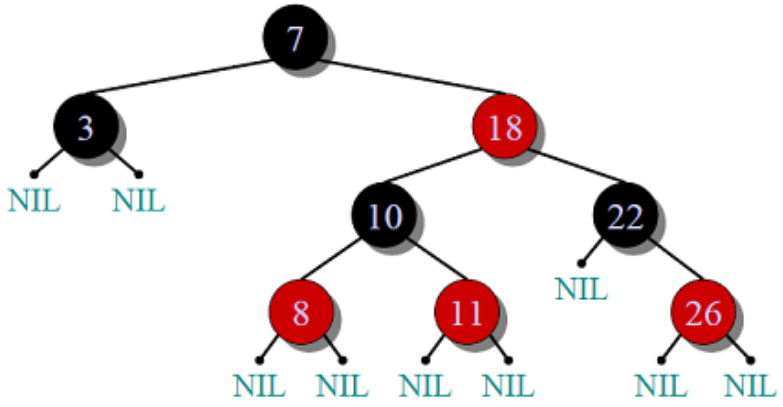
Red-Black Tree | Set 1 (Introduction)

Red-Black Tree is a self-balancing Binary Search Tree (BST) where every node follows following rules.



1) Every node has a color either red or black.

2) Root of tree is always black.

3) There are no two adjacent red nodes (A red node cannot have a red parent or red child).

4) Every path from root to a NULL node has same number of black nodes.

Why Red-Black Trees?
Most of the BST operations (e.g., search, max, min, insert, delete.. etc) take O(h) time where h is the height of the BST. The cost of these operations may become O(n) for a skewed Binary tree. If we make sure that height of the tree remains O(Logn) after every insertion and deletion, then we can guarantee an upper bound of O(Logn) for all these operations. The height of a Red Black tree is always O(Logn) where n is the number of nodes in the tree.

Comparison with AVL Tree
The AVL trees are more balanced compared to Red Black Trees, but they may cause more rotations during insertion and deletion. So if your application involves many frequent insertions and deletions, then Red Black trees should be preferred. And if the insertions and deletions are less frequent and search is more frequent operation, then AVL tree should be preferred over Red

Google™ Custom Search

GeeksforGeeks

52,731 people like GeeksforGeeks.

Interview Experiences
Advanced Data Structures
Dynamic Programming
Greedy Algorithms
Backtracking
Pattern Searching
Divide & Conquer
Mathematical Algorithms
Recursion
Geometric Algorithms

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

Black Tree.

How does a Red-Black Tree ensure balance?
A simple example to understand balancing is, a chain of 3 nodes is not possible in red black tree. We can try any combination of colors and see all of them violate Red-Black tree property.

```
A chain of 3 nodes is nodes is not possible in Red-Black Trees.
Following are NOT Red-Black Trees
        30               30               30
       / \              /  \             /  \
     20  NIL          20   NIL          20   NIL
    / \              / \              /  \
  10  NIL          10  NIL          10  NIL
Violates          Violates         Violates
Property 4.      Property 4       Property 3

Following are different possible Red-Black Trees with above 3 keys
         20                            20
       /   \                         /   \
     10     30                     10     30
    /  \   /  \                   /  \    /  \
 NIL  NIL NIL NIL              NIL  NIL NIL NIL
```

From the above examples, we get some idea how Red-Black trees ensure balance. Following is an important fact about balancing in Red-Black Trees.

Every Red Black Tree with n nodes has height **<=** $2\log_2(n+1)$

This can be proved using following facts:
1) For a general Binary Tree, let **k** be the minimum number of nodes on all root to NULL paths, then n >= 2^k – 1 (Ex. If k is 3, then n is atleast 7). This expression can also be written as k <= $2\log_2(n+1)$

2) From property 4 of Red-Black trees and above claim, we can say in a Red-Black Tree with n nodes, there is a root to leaf path with at-most $\log_2(n+1)$ black nodes.

Popular Posts

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

3) From property 3 of Red-Black trees, we can claim that the number black nodes in a Red-Black tree is at least $\lfloor n/2 \rfloor$ where n is total number of nodes.

From above 2 points, we can conclude the fact that Red Black Tree with **n** nodes has height <= $2\log_2(n+1)$

In this post, we introduced Red-Black trees and discussed how balance is ensured. The hard part is to maintain balance when keys are added and removed. We will soon be discussing insertion and deletion operations in coming posts on Red-Black tree.

Exercise:
1) Is it possible to have all black nodes in a Red-Black tree?
2) Draw a Red-Black Tree that is not an AVL tree structure wise?

Insertion and Deletion
Red Black Tree Insertion
Red-Black Tree Deletion

References:
Introduction to Algorithms 3rd Edition by Clifford Stein, Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest
http://en.wikipedia.org/wiki/Red%E2%80%93black_tree
Video Lecture on Red-Black Tree by Tim Roughgarden
MIT Video Lecture on Red-Black Tree
MIT Lecture Notes on Red Black Tree

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

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

Related Tpoics:

- Print a Binary Tree in Vertical Order | Set 2 (Hashmap based Method)
- Print Right View of a Binary Tree
- Red-Black Tree | Set 3 (Delete)
- Construct a tree from Inorder and Level order traversals
- Print all nodes at distance k from a given node
- Print a Binary Tree in Vertical Order | Set 1
- Interval Tree
- Check if a given Binary Tree is height balanced like a Red-Black Tree

35 Tweet 3 1

Writing code in comment? Please use **ideone.com** and share the link here.

9 Comments GeeksforGeeks

Sort by Newest

695 Subscribe

Recent Comments

affiszerv Your example has two 4s on row 3, that's why it...

Backtracking | Set 7 (Sudoku) · 27 minutes ago

RVM Can someone please elaborate this Qs from above...

Flipkart Interview | Set 6 · 47 minutes ago

Vishal Gupta I talked about as an Interviewer in general,...

Software Engineering Lab, Samsung Interview | Set 2 · 47 minutes ago

@meya Working solution for question 2 of 4f2f round....

Amazon Interview | Set 53 (For SDE-1) · 1 hour ago

sandeep void rearrange(struct node *head) {...

Given a linked list, reverse alternate nodes and append at the end · 2 hours ago

Neha I think that is what it should return as, in...

Find depth of the deepest odd level leaf node · 2

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

Join the discussion…

hours ago

vicky • 2 months ago

In the 3rd point of proof, is it meant that the number of black nodes in a root to half of the total nodes in that path?? If not, then please explain the proof further

Reply • Share ›

Kartik ↗ vicky • a month ago

vicky, we refined the 3rd point. You can refer the video lecture given 3r

Reply • Share ›

dafc • 3 months ago

About the first question, I think a full and complete rb tree with all black nodes i

Reply • Share ›

Saurabh ↗ dafc • 2 months ago

Technically it is possible for the RB tree to have all black nodes. But th all the nodes in the tree using red and black node combination and ther to black. Because if the tree is completely black then any new leaf nod break the RB tree property of all leaf node have same black height.

Reply • Share ›

Kartik ↗ dafc • 3 months ago

Agree, a full and complete tree can only have all black nodes

Reply • Share ›

Ronny • 3 months ago

How is the second example violating rule number 1. Rather it should be rule nu

30(B)
/ \

AdChoices
▶ Binary Tree
▶ Java Tree
▶ Java to C++
AdChoices
▶ Red Black Tree
▶ Java Array
▶ Tree Balancing
AdChoices
▶ Tree Trees
▶ Tree Full
▶ Tree Structure

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

/ \
10(B) NIL

• Reply • Share ›

GeeksforGeeks Ronny • 3 months ago

Thanks for pointing this out. We have updated the example.

• Reply • Share ›

dzanjo • 3 months ago

20
/ \
10 30
/ \ / \
NIL NIL NIL NIL
Isn't it voilating property no 4?

• Reply • Share ›

GeeksforGeeks dzanjo • 3 months ago

Thanks for pointing this out. We have removed the incorrect tree from

• Reply • Share ›

Subscribe Add Disqus to your site

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com

@geeksforgeeks, **Some rights reserved** **Contact Us!** Powered by **WordPress** & **MooTools**, customized by geeksforgeeks team

open in browser PRO version Are you a developer? Try out the HTML to PDF API pdfcrowd.com