**CS430 Lecture 11 Activities**

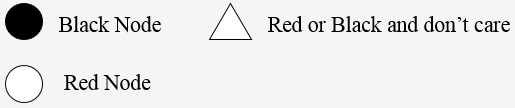
Opening Questions

1. What do you think the issue we need to handle when deleting a node from a red-black tree? How does red-black delete differ from a BST delete?

Red-Black Tree Delete  
Think of V as having an “extra” unit of blackness. This extra blackness must be absorbed into the tree (by a red node), or propagated up to the root and out of the tree. There are four cases – our examples and “rules” assume that V is a left child. There are symmetric cases for V as a right child

Terminology in Examples

* The node just deleted was U
* The node that replaces it is V, which has an extra unit of blackness
* The parent of V is P
* The sibling of V is S



|  |  |
| --- | --- |
| * V’s sibling, S, is Red   + Rotate S around P and recolor S & P * NOT a terminal case – One of the other cases will now apply * All other cases apply when S is Black |  |
| * V’s sibling, S, is black and has two black children.   + Recolor S to be Red   + P absorbs V’s extra blackness     - If P was Red, make it black, we’re done     - If P was Black, it now has extra blackness and problem has been propagated up the tree |  |
| * S is black * S’s RIGHT child is RED (Left child either color)   + Rotate S around P   + Swap colors of S and P, and color S’s Right child Black * This is the terminal case – we’re done |  |
| * S is Black, S’s right child is Black and S’s left child is Red   + Rotate S’s left child around S   + Swap color of S and S’s left child   + Now in case 3 |  |

Red Black Visualization - <http://gauss.ececs.uc.edu/RedBlack/redblack.html>

AVL Trees

An AVL tree is a special type of binary tree that is always "partially" balanced. The criteria that is used to determine the "level" of "balanced-ness" is the difference between the heights of sub-trees of every node in the tree. The "height" of tree is the "number of levels" in the tree.

An AVL tree is a binary tree in which the difference between the height of the right and left sub-trees (of any node) is never more than one.

1. How do you think we could keep track of the height of the right and left sub-trees of every node?

2. If we find an imbalance, how can we correct it without adding any significant cost to the insert or delete?

|  |  |
| --- | --- |
| Single Rotations  The imbalance is left-left (or right-right)    Perform single right rotation at “c” (R-rotation)  Similar idea for single left rotation (L-Rotation) | Double Rotations  The imbalance is left-right (or right-left)    Perform right rotation at “c” then left rotation at “a” (RL-rotation)  Similar idea for left rotation then right rotation (LR-Rotation) |

AVL Visualization <http://www.cs.umd.edu/class/spring2002/cmsc420-0401/demo/avltree/>