Pawan (PK) Khatri, Pawan Chandra

CSC 130

Chidella

11 June 2018

Project #2 Write-Up

The purpose of this project was to implement and experiment a data structure known as Binary Search Tree (BST). The BST we created was randomized that guarantees to have an expected height of 5 without performing any tree balance operations. The BST we created is a special BST which uses a *randomized* version of the *insertion* and *deletion* operations. For the insertion, in each node we stored the size of the subtree rooted at that node.

For the AVL tree class, we used the methods discussed in the PowerPoint to the rotations for singleRightRotation (Case 1), doubleLeftRightRotation (Case 2), singleLeftRotation (Case 3), and doubleRightLeftRotation (Case 4).

From doing the comparisons in this project, we determined that the AVL tree makes it faster in worst cases. It keeps rebalancing itself, so in worst case it will consume O(log n ) time when the plain BST will take O(n). AVL has 3 main properties, insertion, deletion, and balancing. These properties make the AVL tree great to use especially if we have large data. It does however, require few more steps than a regular BST when the difference in height is greater than 1. In conclusion, we determined that it is always better to implement an AVL tree as its faster to read from and implement or remove nodes without sacrificing time and space.