**CH320201-Algorithms and Data Structures**

**Spring 2019**

**Red Black Trees**

**Homework 8**

**Problem 8.1**

***Understanding Red Black Trees***

1. **(3 points) Draw (or describe by using preorder traversal) the red-black trees that result after successively inserting the values step by step in the following order [13, 44, 37, 7, 22, 16] into an empty red-black tree. You are required to draw (or describe by using preorder traversal) the tree after each insertion, as well as any additional recoloring and balancing.**

To insert a node into a red black tree, there are different conditions that must be checked and violations must be resolved. To insert an element, you add a node Z, initially as the root of the tree and build on the left and right sub-properties. Every new node entered in painted red at first, and violations are then sorted using recoloring and rotations.

13 is inserted at the root, followed by 44 which is the right child of the root. 37 is then inserted as the left child of 44, but then violations are sorted and 37 is now the root with 14 as the left child and 44 as the right child of the root. Both children are red, while the root is black at this time.

7 is inserted as the left child of 13, making the root its grandparent and 44 its uncle. At this point, only 7 is a red node and the rest are black. At this point 22 is inserted as the right child of 13 and is also a red node.

Finally 16 is inserted as the left child of 22 and as a result, 22 is now a black node and so is 7. 13 is a red node after recoloring and so are 44 and 37.

The Red Black Tree has been constructed with all conditions fulfilled.

1. **(3 points) Draw (or describe by using preorder traversal) all valid red-black trees that store the values {1, 2, 3, 4}.**

Inserting {1, 2, 3, 4} will have at most 24 possible permutations in which they can be inserted into a tree. Out of these, only 4 trees shall be valid, and is such noted on the following pages.

1. **(3 points) Bonus: Consider a red-black tree formed by inserting n nodes with the algorithm described in the lecture slides. Prove that if n > 1, the tree contains at least one red node.**

This proof can be done using highlighting the possible cases as well as using induction. I have shown both, and have used some online forums in understanding the proof initially.