Name: Sang Hun Kim
PUID: 0025808690
3/28/2017

Programming Assignment 4 Report

From this assignment, it was asked to compute for the minimum size of rectangle for packing. The packing that was done for programming assignment 3 is one of the 2n-3 ways that binary tree can be re-rooted, in which ‘n’ in 2n-3 is the leaf nodes existing in the binary tree.

According to the code for performing re-root, the time complexity for finding all the possible way to find the minimum size of rectangle is O(n). In order for the program to find the minimum size of rectangle for packing, it is necessary to visit all the nodes that does not contain path, in which it is defined as H and V. Pre-order traversal fashion of re-rooting the original tree has been performed in the program. To explain further, re-rooting function first locates its root of the tree and determines whether its children nodes do have path. If it does, then starting from the left side, it performs the re-root by reorganizing children nodes. But here, the program actually does not modify its original shape, but instead, it calculates for the new width and new height of the packed rectangle after re-root. In this way, re-root function traverses all the nodes with path (does not enter into the leaf node) and searches for the minimum size of the packed rectangle. Here, because there are n number of leaf nodes, time complexity for searching for all the method will exactly be O(2n-3); however in general, we can conclude that its time complexity is O(n).

The space complexity for this re-root performance is roughly O(log(n)) because it is to be known that the height of the binary tree is roughly log(n). This represents that the number of stack frames that exist for executing re-root is same is the height of the tree. Thus, it can be concluded that the space complexity for executing re-root is O(log(n)).