Allen Proulx | Programming Assignment 3

For constructing the binary tree, the input file data has to be first read in and then constructed into a tree. To read in the data for each node I first create a node on heap from the file data then add this to the top of a stack of nodes. So this is O(n) operations because I’m creating a node for each element in the file. Next, from this stack of nodes I create the tree. I do this by popping a node from the stack and linking the pointers from the parent to the popped node. This way I am only at each node when it is created. Therefore, the time complexity of building the tree is O(n). So the total time complexity of building the tree from the input file is O(2n).

Computing the coordinates of each node requires computing the coordinates of all the nodes under it and to its left. This requires that the coordinates of each node be assigned from the most left and lowest node to the highest and most right nodes. I do this by using recursion to calculate the coordinates as each function call returns. At every H node I calculate the coordinates right then left and at every V node I calculate left then right. This allows me to only have to visit each node once. However, if there are H or V nodes to the left of an H node or the right of a V node, the coordinates are incorrect because the calculation does not build up and therefore does not know that there are node under it or to its left. However, to fix this, in each call as I’m traversing down the tree, I send an offset value that accounts for the nodes below and or to the left. This keeping the programming from returning to nodes to recalculate the coordinates so that they are correct. Doing this keeps the time complexity of packing to O(n).