Zixian Lai

ECE 36800

Report

The analysis of the space and time complexity of the four non-recursive tree:

The space complexity for the postorder ,preorder and inorder non-recursive tree traversals would all be O(h) (h is the depth of the binary tree), since I am only using on stack to implement the function. And the time complexity would be O(m + n) which can be finalized in to O(n), too.

The time complexity for the Breadth first search would be O(n^2) since each node will donate two child node and n is the number of total number of nodes in the tree. The space complexity for the breadth first search would be O(n) since the function save the whole tree in the stack and pop one by one.

The tabulation of the number of comparisons for three different search case:

The best case for all three case would be the key is the head of the tree or list, it will only one comparison to find it, time complexity = O(1). The worst for the list and the unbalanced tree would be all the elements are identical and searching for a different element. Then the function would search all the nodes to prove that the element doesn't exit with time complexity O(n). The worst case for the balanced tree is that the element at the bottom of the tree which will take O(logn) to find it.

Summary: It will take O(n) to make the linked list, O(nlogn) to build the tree from the linked list, and O(nlogn) to build the balanced tree, too. Because instead of balance the unbalanced tree I made two new pointers and built a new balanced tree from list.