cs5008-hw-bst

Name: Minjie Shen

1. What does it mean if a binary search tree is a balanced tree?

A balanced binary tree is defined as a binary tree in which the height of the left and right subtree of any node differs by not more than 1. The height of the node is the level number of the tree. For example, if the deepest root node of the left subtree is in level 5, and the deepest root node of the right subtree is in level 4, this binary search tree is called a balanced tree since the difference between the height of the left and the right subtree differs by no more than 1. However, if the deepest root node of the right subtree is in level 3, this is not a valid balanced tree anymore.

2. What is the big-Oh search time for a balanced binary tree? Give a logical argument for your response. You may assume that the binary tree is perfectly balanced and full.

The search time for a balanced binary tree is O(LogN), and the N represents the total number of nodes in this BST. If we want to search a node with value X, we first compare the value X with the root node value, if the value X is less than the root node value, we know that the node of value X must be in the left subtree of the BST, and we can eliminate the whole right subtree nodes, which is close to N/2 nodes (since the binary try is perfectly balanced and full). By this logic, we can eliminate half of the total possible nodes in each search time. If in the worst case, the value X node is in the leaf of the BST, we only need LogN times to find the target node.

3. Now think about a binary search tree in general, and that it is basically a linked list with up to two paths from each node. Could a binary tree ever exhibit an O(n) worst-case search time? Explain why or why not. It may be helpful to think of an example of operations that could exhibit worst-case behavior if you believe it is so.

Yes, a binary tree will exhibit an O(N) search time in the worst case. For example, if the root of the binary search tree is 1, and all other nodes are squeezed on one side of the root, for instance, the next node is 2, which is on the right node of root 1, and the next node is 3, which is on the right node of node 2, etc. In this situation, the binary tree will look like a line with height N. If the target node we want to find is in the leaf of this binary tree, we will take O(N) time to search that node.