Data Structures
Binary Search Trees

Cain Susko

Queen's University
School of Computing

February 6, 2022

Binary Search Tree

The main purpose of this type of tree is for searching. It is extremely efficient at searching and storing data. You will likely be asked about this in job interviews.
A binary search tree is a Binary tree with unique properties:

- for every node x in a tree:
  - all nodes in x's left subtree must be *less* than x
  - all nodes in x's right subtree are *greater* than x

A tree is a binary search tree if and only if it is a binary tree and it satisfies these conditions.
Many algorithms will specify that duplicates are excluded and that the left children are *less than or equal to* x

Validation

the algorithm for checking the validity of a binary search tree is to go through each node and check if the subtrees match the conditions above. Furthermore, while this isn't necissarily unique, printing the read-order of Inorder traversal of a biary search tree will print out the items in order from highest to lowest.

Operations

the operations on a binary search tree are:

- search
- insert
- delete
- traverse

Search

Search is the most important operation for binary search trees. The algorithm is as follows:
Given a target value v

1. start at the root of the tree and process unitll we hit a None object.
2. if v is less than the current nodes value, go left
3. if v is equal to the current nodes value, return true
4. if v is greater than the current nodes value, go right
5. if the current nodes value is None, return false

It is possible to write this algorithm both iteratively and recursively.
Recursively, we check a value and then recursively call the check function again on the left or right node. Iteratively, we iterate through the nodes in the same manner. In the real world, the iterative method is faster. but with regards to big O, the recursive method is better.
generally the worst case big O for binary tree search is:

$$O(n).$$

Which is incredible for a worst case evaluation.
Note that the Big O complexity is

$$\log_2(n).$$