Binary Search Tree (BST) Finale

Q: How does our data determine the height?

1 3 2 4 5 7 6  vs.  4 2 3 6 7 1 5

Q: How many different ways are there to insert data into a BST?

Q: What is the average height of every arrangement?

....what’s the intuition for this argument?

<table>
<thead>
<tr>
<th>operation</th>
<th>BST Avg. Case</th>
<th>BST Worst Case</th>
<th>Sorted Array</th>
<th>Sorted List</th>
</tr>
</thead>
<tbody>
<tr>
<td>find</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>insert</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>delete</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>traverse</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Let us describe the balance (b) of a BST to be:

- If b is negative:
- If b is positive:

We define a BST tree T to be **height balanced** if:

A node is considered to be **out of balance** if it’s not height balanced. What is the lowest node that is out of balance?
Brining a tree back into balance
Goal: Create a strategy to bring a BST back into balance after an operation has caused the tree to be out of balance.

A Tree Rotation is an operation that maintains two properties:

1. 
2. 

Example 1: Defining a Rotation

1. Where is the deepest point of imbalance in the tree: ➔

2. Perform a left rotation to balance this tree:

Implementing a left rotation:

Example 2: A Complex Rotation

Rotation #1:

Rotation #2:

BST Rotation Summary:

1. Four kinds of rotations (L, R, LR, and RL)
2. All rotations are local
3. All rotations run in constant time, O(1)
4. BST property is maintained!

Overall Goal:

...and we call these trees:

CS 225 – Things To Be Doing:

1. Theory Exam 2 starts next Thursday (topic list is online)
2. MP3 due **tonight**; MP4 released on Tuesday
3. lab_huffman released Wednesday
4. Daily POTDs