CS 225
Data Structures

September 20 – BST Balance
G Carl Evans
<table>
<thead>
<tr>
<th>Operation</th>
<th>BST Average 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>
BST Analysis

Lower bound:

Upper bound:
Height-Balanced Tree

What tree makes you happier?

Height balance: \( b = \text{height}(T_R) - \text{height}(T_L) \)

A tree is height balanced if:
BST Rotation

We will perform a rotation that maintains two properties:

1.

2.
AVL Tree Rotations

Four templates for rotations:
AVL Trees

Three issues for consideration:
- Rotations
- Maintaining Height
- Detecting Imbalance
BST Rotation Summary

- Four kinds of rotations (L, R, LR, RL)
- All rotations are local (subtrees are not impacted)
- All rotations are constant time: O(1)
- BST property maintained

GOAL:

We call these trees: