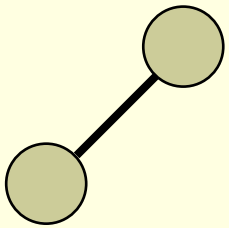
AVL Trees: Insertion Revisited

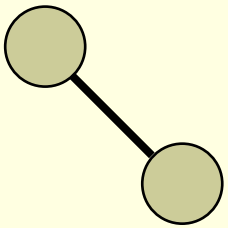
Computer Science Department
University of Central Florida

COP 3502 – Computer Science I

Insertion Revisited

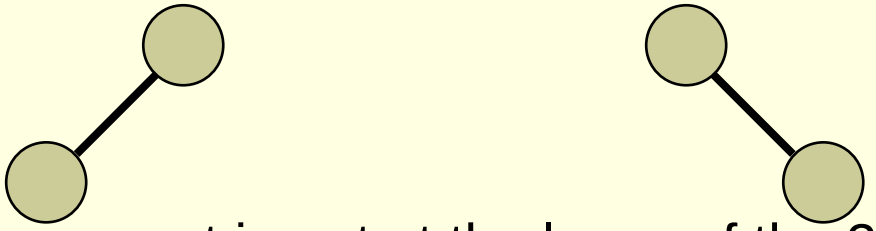
- AVL Trees: Insertion
  - Let's take another look at insertion into AVL Trees
  - Hopefully this will be a bit easier than previous slides
  - Assuming you only have two nodes in your tree,
  -





AVL Trees: Insertion Revisited

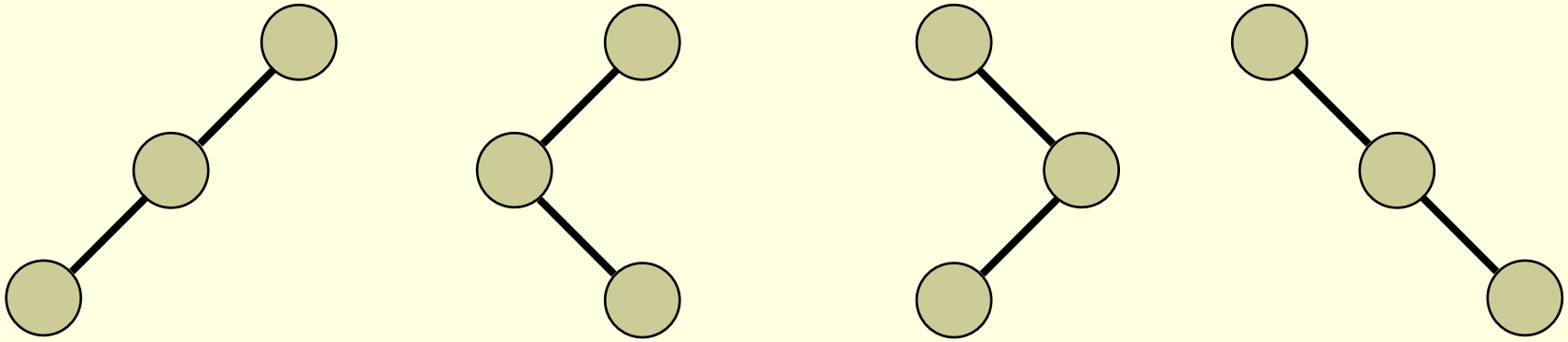
- AVL Trees: Insertion
  - Given these two trees, if we want to create an imbalance, where must we insert?



    - Clearly, we must insert at the lower of the 2 nodes
    - This will create a scenario where the left subtree has a height that is 2 greater than the right subtree
      - Or the opposite for the other tree depicted

AVL Trees: Insertion Revisited

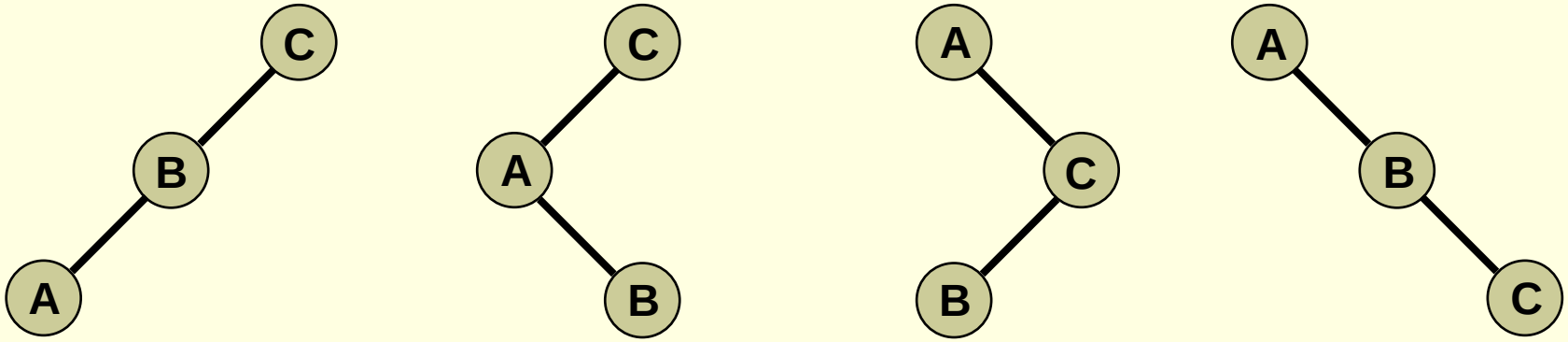
- AVL Trees: Insertion
  - Here are all four unbalanced trees that we can make from three nodes:



    - Now, label these nodes with the labels A, B, and C
      - Where A is the smallest of the three nodes, B is the middle node, and C is the largest.

AVL Trees: Insertion Revisited

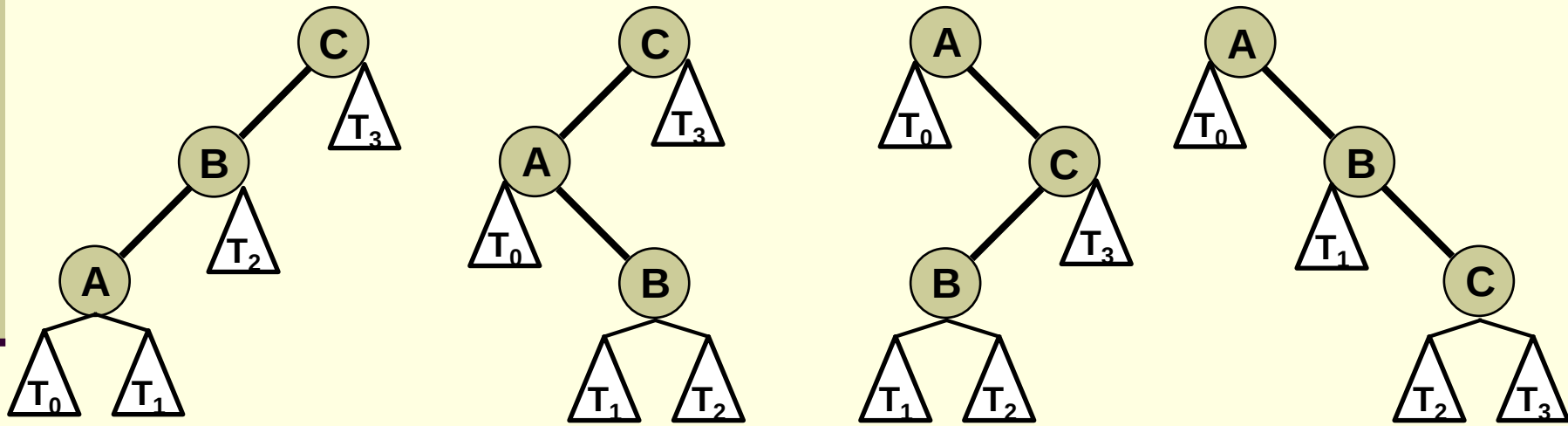
- AVL Trees: Insertion
  - Here are all four trees with the node lables in their inorder listing:



    - Any time an imbalance occurs, it is localized to three nodes and their four subtrees
      - These are the four possibilities
      - Now we add in the depiction of the four subtrees

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Here are all four trees with the node lables in their **inorder** listing with subtrees in their **inorder** listing:



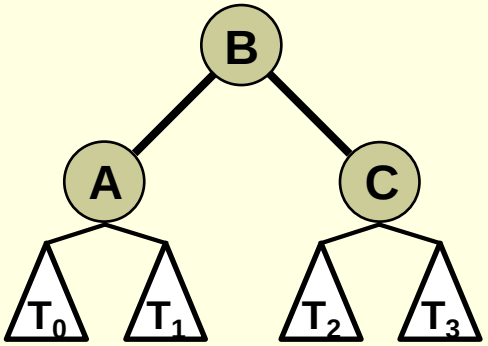
  - We denote the four subtrees as T_0, T_1, T_2, and T_3

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - So what is the purpose of all this?
  - We said this method is supposedly MUCH easier than dealing with the various rotations of the tree
  - So we've done all this labeling
    - Finding nodes 'A', 'B', and 'C' and labeling them as such
  - How they heck does this help us???

AVL Trees: Insertion Revisited

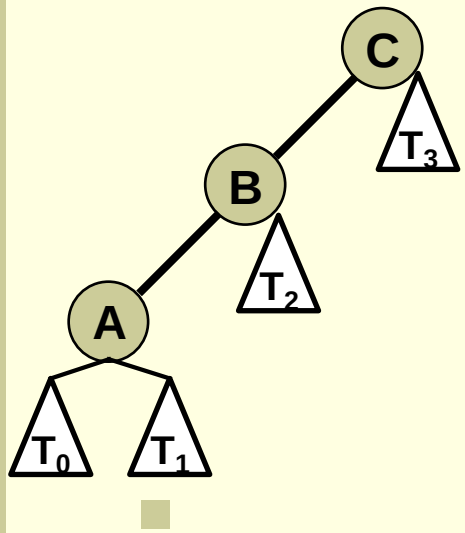
- AVL Trees: Insertion
  - Part 1: Once an insertion causes an imbalance, find and label the nodes ‘A’, ‘B’, and ‘C’
  - Part 2: Once the nodes are labeled, no matter what structural imbalance occurred, they can all be fixed the same way:

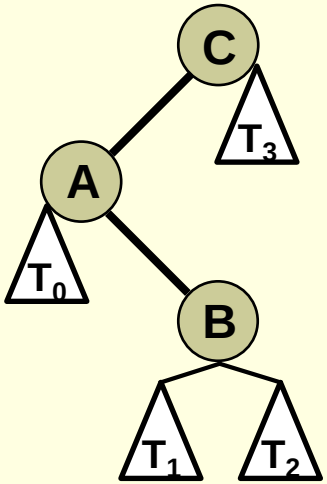


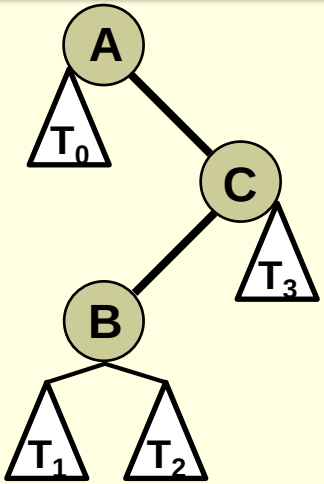
AVL Trees: Insertion Revisited

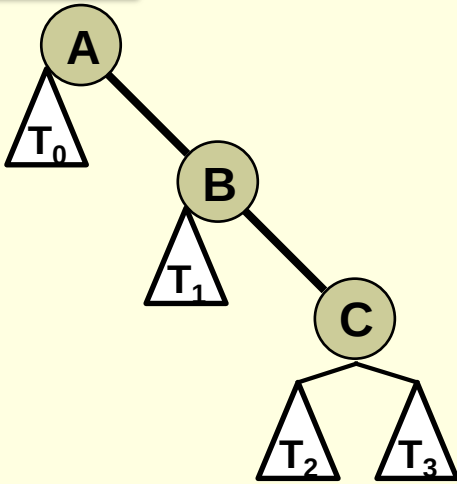
- AVL Trees: Insertion

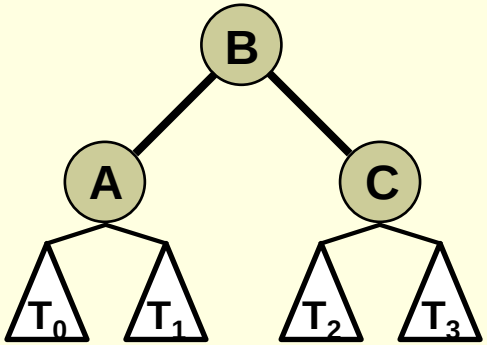
All 4 of these trees:









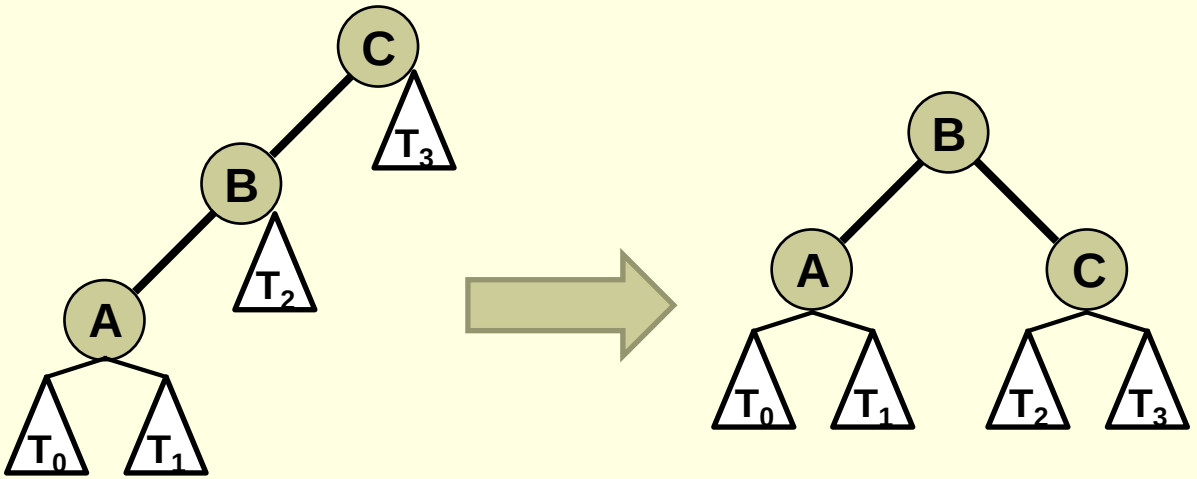


Are we really doing any rotations!

- Okay, let's go back and see 4 scenarios
- 1) insertion into the left subtree of the left child of the root.
- 2) insertion into the right subtree of the left child of the root.
- 3) insertion into the left subtree of the right child of the root.
- 4) insertion into the right subtree of the right child of the root

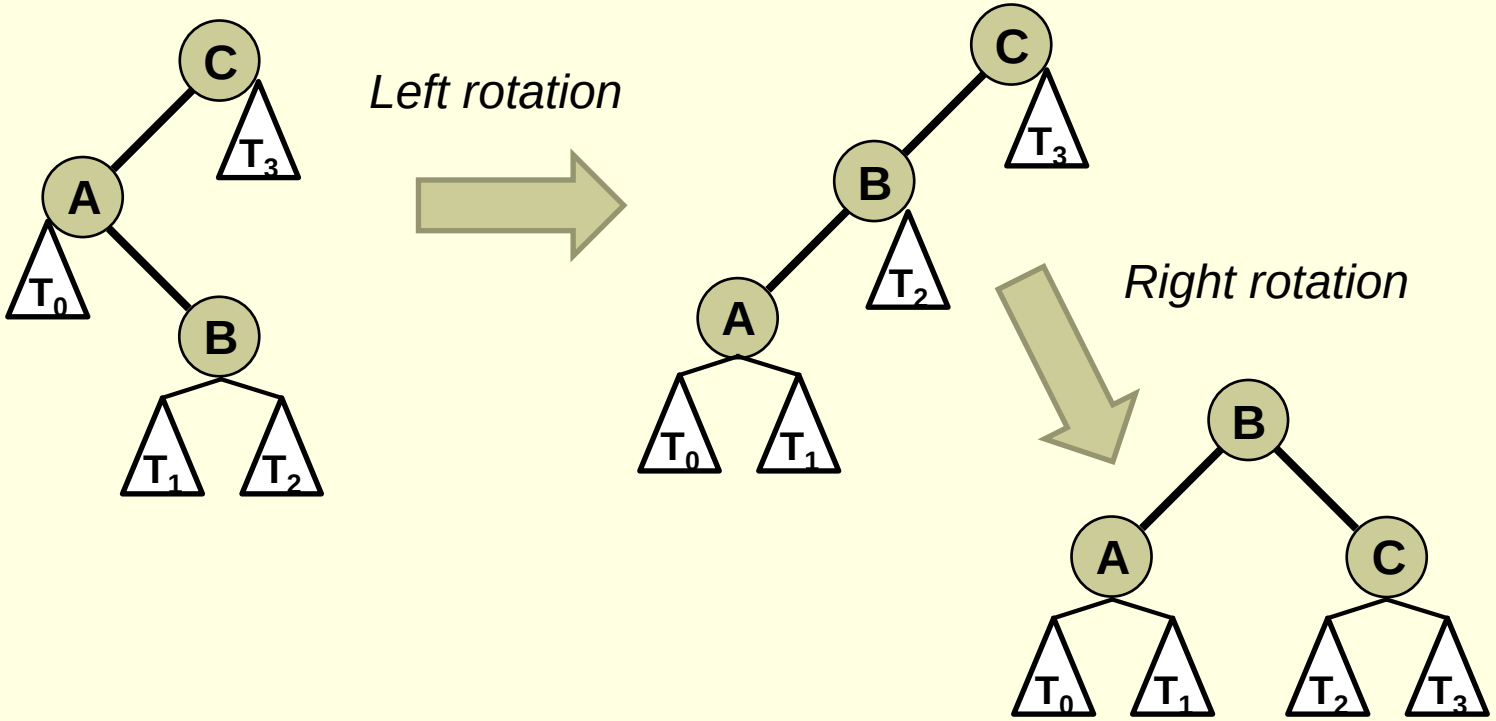
Four Scenarios (1)

- (1) insertion into the left subtree of the left child of the root.
- The following figure represents this.
- In this case we are doing only one rotation (right rotation)



Four Scenarios (2)

- (2)insertion into the right subtree of the left child of the root. The following figure represents this.
- In this case we are doing **double** rotations

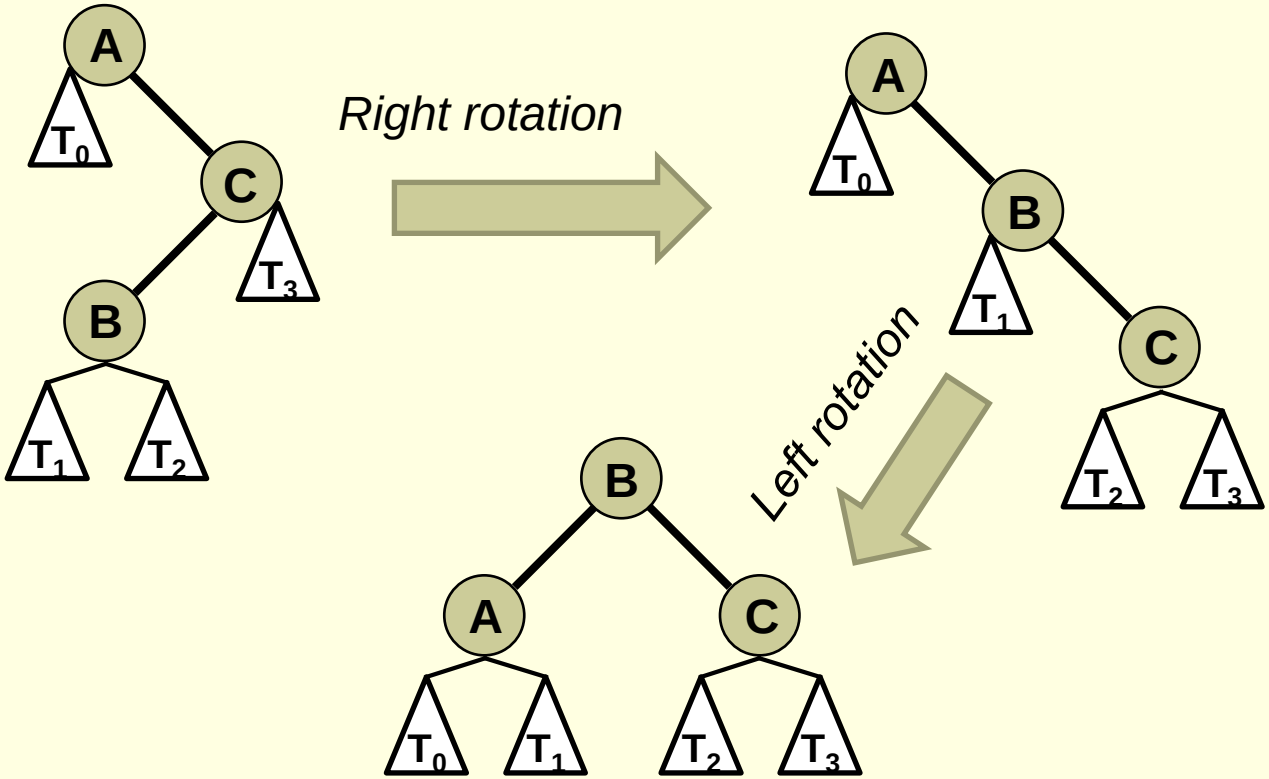


Four Scenarios (3)

- (3) insertion into the left subtree of the right child of the root. (Symmetric to scenario 2).

The following figure represents this.

- In this case we are doing **double** rotations

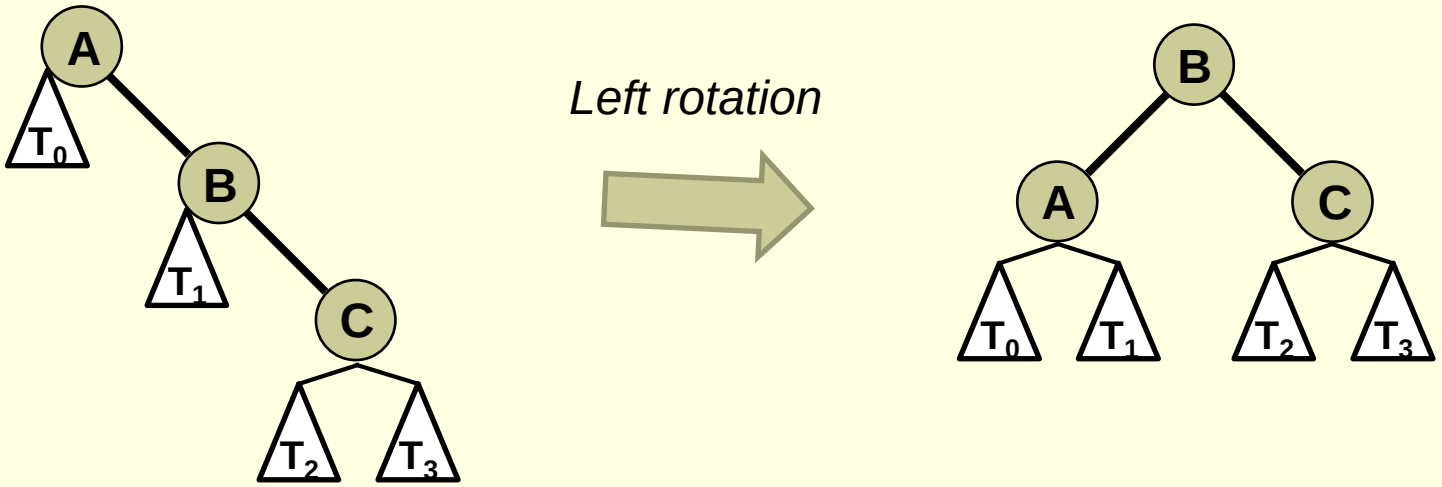


Four Scenarios (4)

- (4) insertion into the right subtree of the right child of the root (Symmetric to scenario 1).

The following figure represents this.

- In this case we are doing **double** rotations



AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Here are the basic steps:

1. Do a NORMAL binary search tree insert
   - following the ordering property of a BST
2. Restore the balance of the tree (if needed) based off of this newly inserted leaf node

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Restoration of a node:
    1. Calculate the heights of the left and right subtrees. Use this to set the (potentially) new height of the node
    2. IF they are within one of each other, recursively restore the parent node.
    3. IF NOT, then perform the appropriate restructuring, described previously, on that particular node.
    4. THEN, recursively call the restore function on the appropriate parent node.

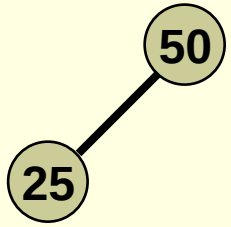
AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - More **Practical Rules**:
    - Insert a node following rules of BST insertion
    - Once you insert a new node, perform the following:

1. Start finding the balance factors of ALL nodes along the path from the insertion point to the root
2. As soon as you find the first node out of balance, mark that node as one of your three "restructuring nodes"
3. Then, take two steps, back down, towards the insertion point and mark those two nodes as well.
4. Label those 'A, B, C' nodes appropriately (and subtrees)
5. Restructure those three nodes (and their subtrees)

AVL Trees: Insertion Revisited

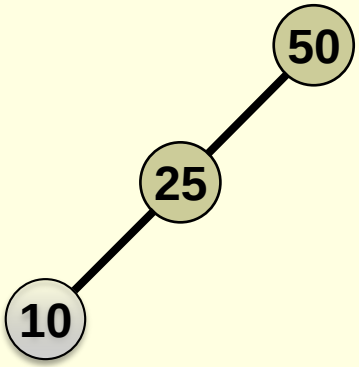
- AVL Trees: Insertion
  - Example 1:
    - The most simple insert into an AVL tree, which dictates a rebalance, is inserting a third node in an AVL tree that only has two nodes.
    - Given this tree:



      - We insert a node with the value 10
      -

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Example 1:
    - The most simple insert into an AVL tree, which dictates a rebalance, is inserting a third node in an AVL tree that only has two nodes.



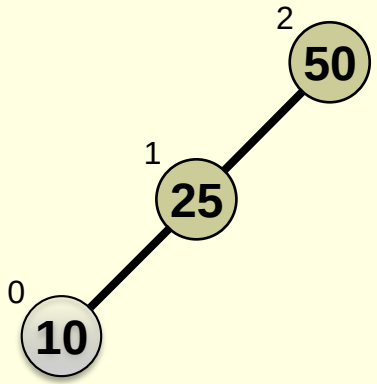
AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - More **Practical Rules**:
    - Insert a node following rules of BST insertion
    - Once you insert a new node, perform the following:

1. Start finding the balance factors of ALL nodes along the path from the insertion point to the root
2. As soon as you find the first node out of balance, mark that node as one of your three "restructuring nodes"
3. Then, take two steps, back down, towards the insertion point and mark those two nodes as well.
4. Label those 'A, B, C' nodes appropriately (and subtrees)
5. Restructure those three nodes (and their subtrees)

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Example 1:

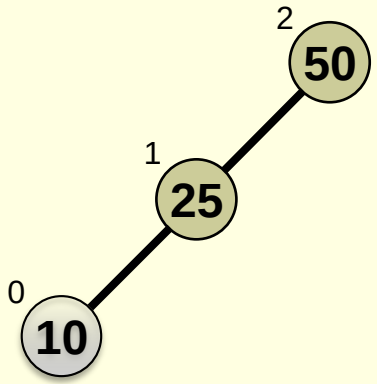


    - So now, follow the “Practical Rules” to rebalance this tree

    1. Start finding the balance factors of ALL nodes

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Example 1:

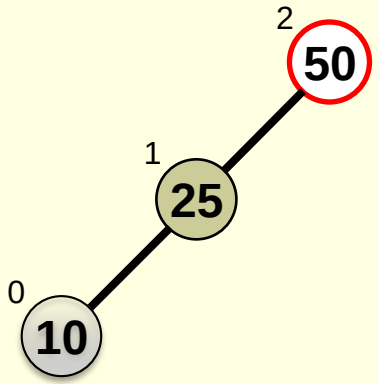


    - So now, follow the “Practical Rules” to rebalance this tree

    2. As soon as you find the first node out of balance, mark that node

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Example 1:

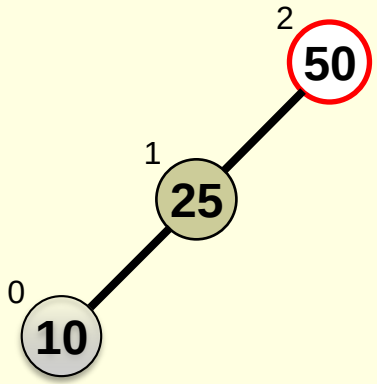


    - So now, follow the “Practical Rules” to rebalance this tree

    2. As soon as you find the first node out of balance, mark that node as one of your three “restructuring nodes”

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Example 1:

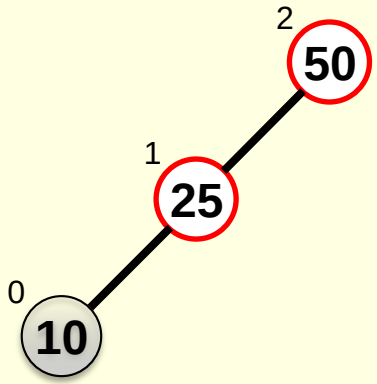


    - So now, follow the “Practical Rules” to rebalance this tree

    3. Then, take two steps, back down, towards the insertion point and mark those two nodes

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Example 1:

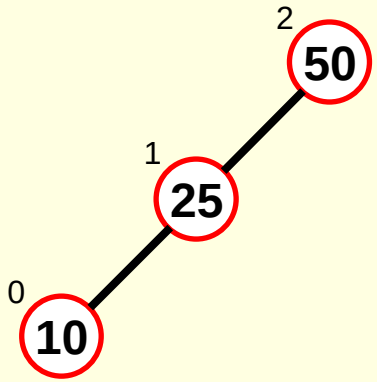


    - So now, follow the “Practical Rules” to rebalance this tree

    3. Then, take two steps, back down, towards the insertion point and mark those two nodes as well.

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Example 1:

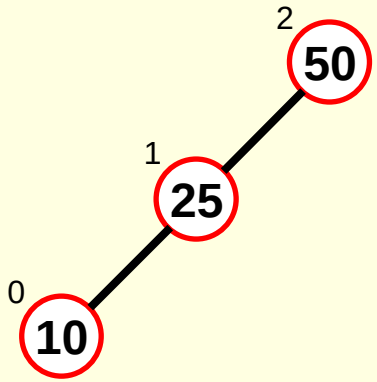


    - So now, follow the “Practical Rules” to rebalance this tree

    3. Then, take two steps, back down, towards the insertion point and mark those two nodes as well.

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Example 1:

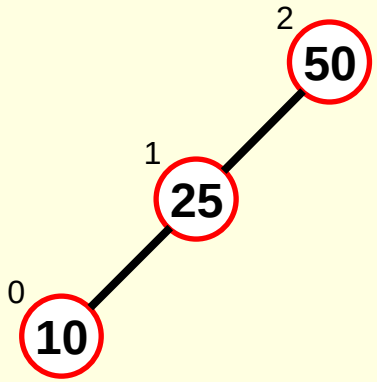


    - So now, follow the “Practical Rules” to rebalance this tree

    3. Then, take two steps, back down, towards the insertion point and mark those two nodes as well.

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Example 1:

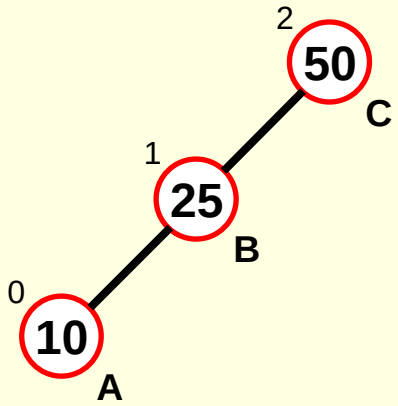


    - So now, follow the “Practical Rules” to rebalance this tree

4. Label those ‘A, B, C’ nodes appropriately (and subtrees)
   - Remember, of the three nodes:
   - The smallest node should be labeled ‘A’
   - The middle node should be labeled ‘B’
   -

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Example 1:

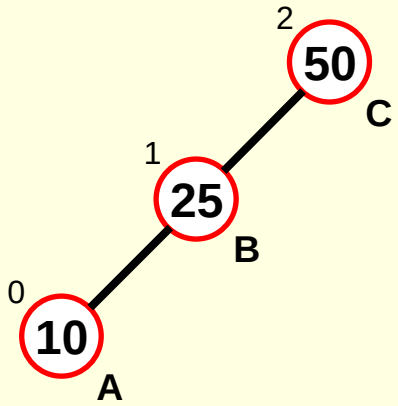


    - So now, follow the “Practical Rules” to rebalance this tree

4. Label those ‘A, B, C’ nodes appropriately (and subtrees)
   - Remember, of the three nodes:
   - The smallest node should be labeled ‘A’
   - The middle node should be labeled ‘B’
   - The largest node should be labeled ‘C’

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Example 1:



    - So now, follow the “Practical Rules” to rebalance this tree

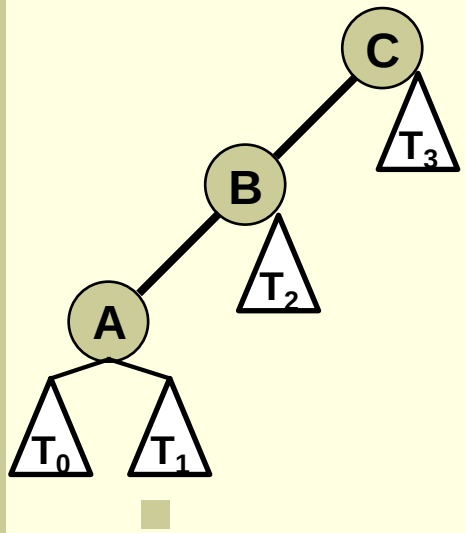
    5. <u>Restructure</u> those three nodes (and their subtrees)

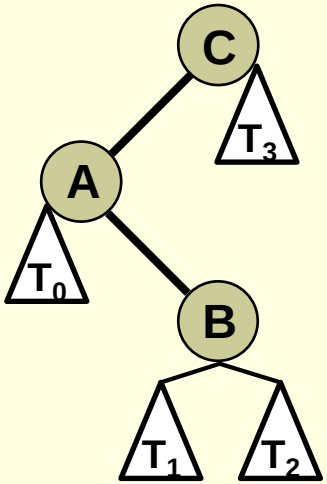
    ■

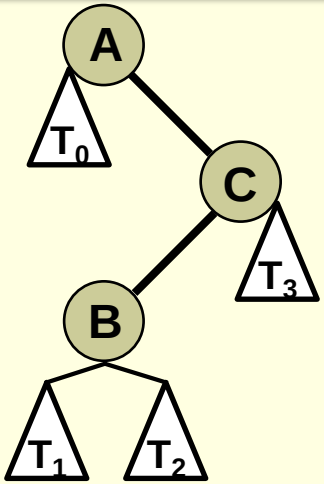
AVL Trees: Insertion Revisited

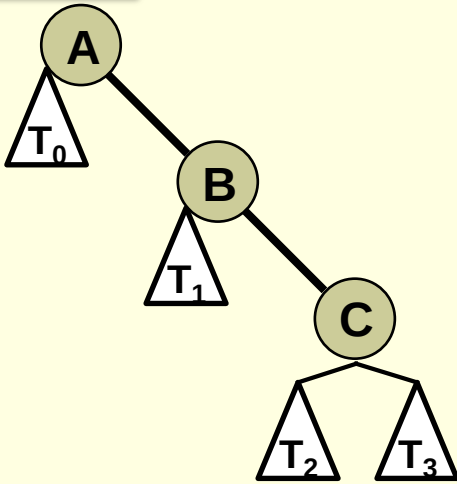
- AVL Trees: Insertion

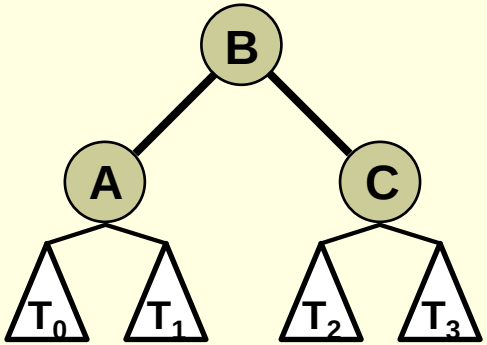
All 4 of these trees:





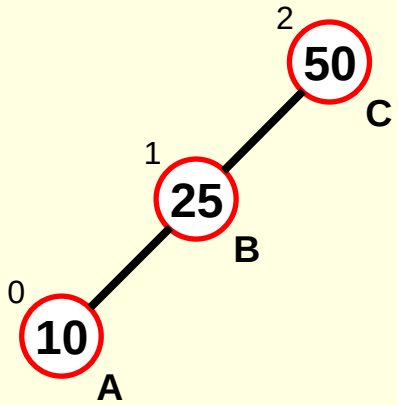






AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Example 1:

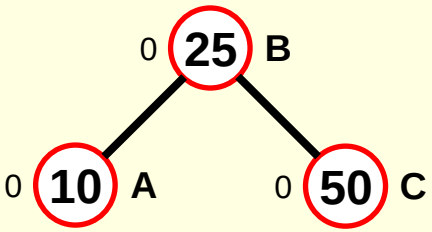


    - So now, follow the “Practical Rules” to rebalance this tree

    5. <u>Restructure</u> those three nodes (and their subtrees)

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Example 1:



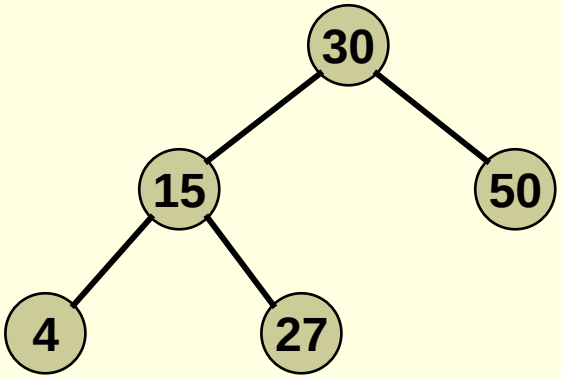
- So now, follow the “Practical Rules” to rebalance this tree

5. Restructure those three nodes (and their subtrees)

   - So we simply restructure the tree according to the previous slide

AVL Trees: Insertion Revisited

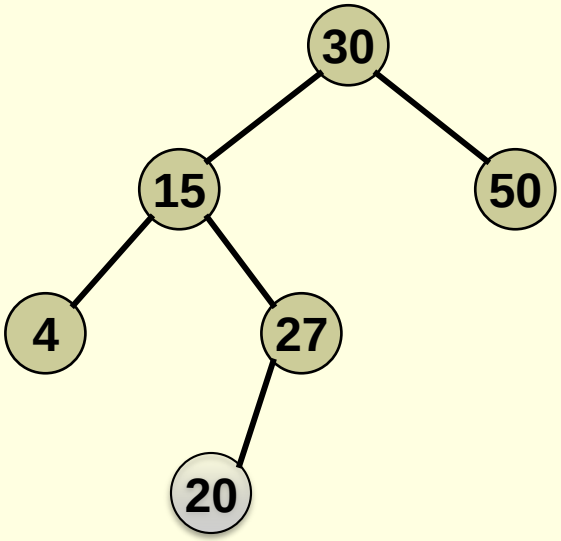
- AVL Trees: Insertion
  - Example 2:
    - Given this tree:



      - We insert a node with the value 20
      -

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Example 2:
    - Given this tree:



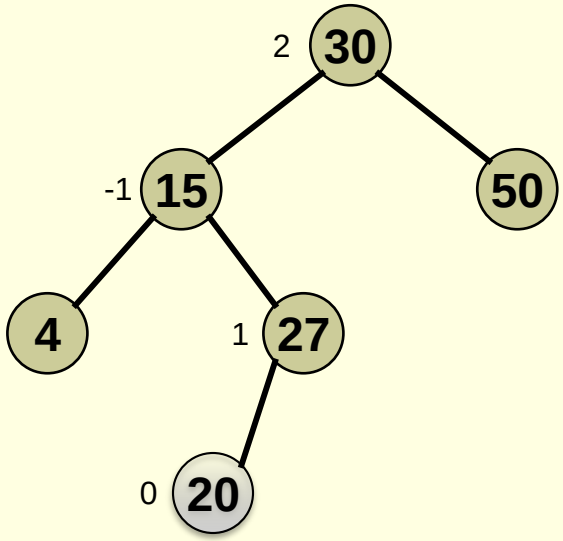
AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - More **Practical Rules**:
    - Insert a node following rules of BST insertion
    - Once you insert a new node, perform the following:

1. Start finding the balance factors of ALL nodes along the path from the insertion point to the root
2. As soon as you find the first node out of balance, mark that node as one of your three "restructuring nodes"
3. Then, take two steps, back down, towards the insertion point and mark those two nodes as well.
4. Label those 'A, B, C' nodes appropriately (and subtrees)
5. Restructure those three nodes (and their subtrees)

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Example 2:

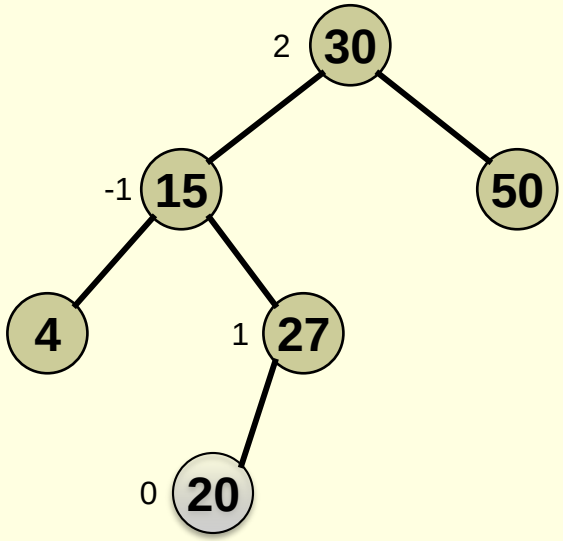


    - So now, follow the “Practical Rules” to rebalance this tree

    1. Start finding the balance factors of ALL nodes

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Example 2:

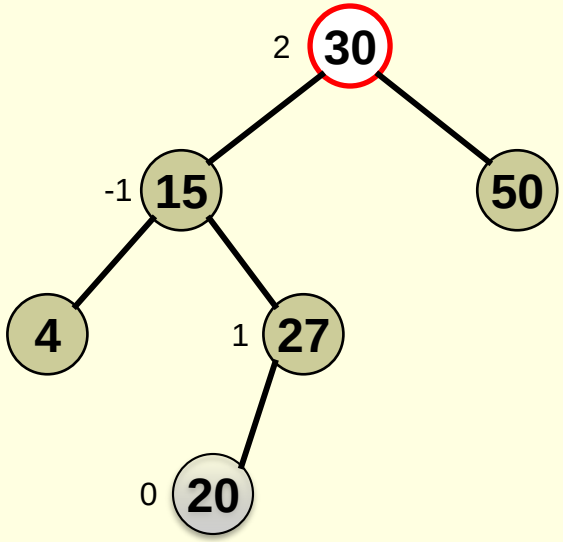


    - So now, follow the “Practical Rules” to rebalance this tree

    2. <u>As soon as</u> you find the first node out of balance, <u>mark that node</u>

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Example 2:

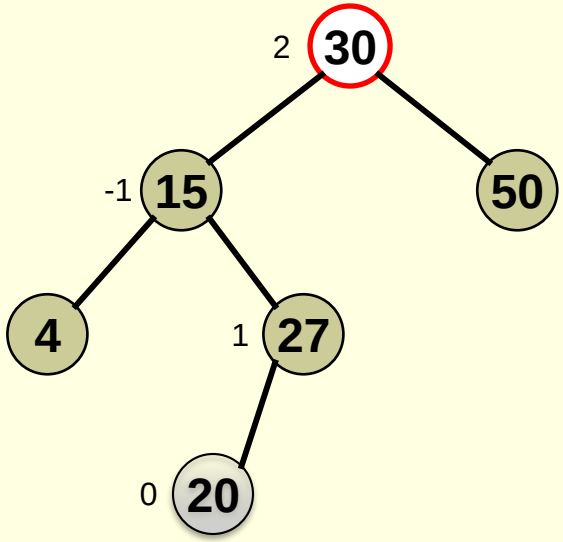


    - So now, follow the “Practical Rules” to rebalance this tree

    2. As soon as you find the first node out of balance, mark that node as one of your three “restructuring nodes

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Example 2:

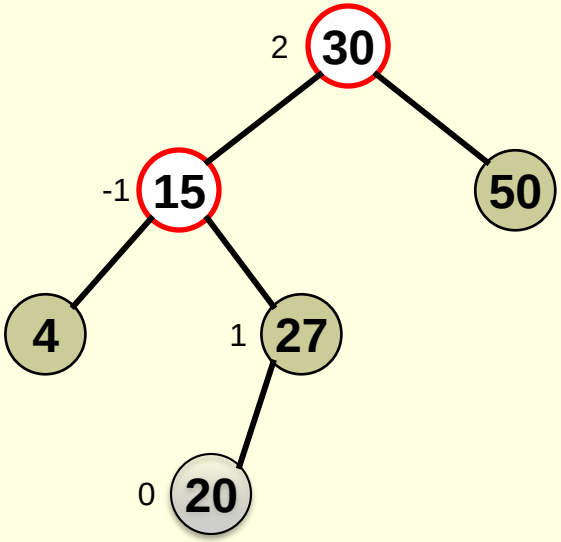


    - So now, follow the “Practical Rules” to rebalance this tree

    3. Then, take two steps, back down, towards the insertion point and mark those two nodes

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Example 2:

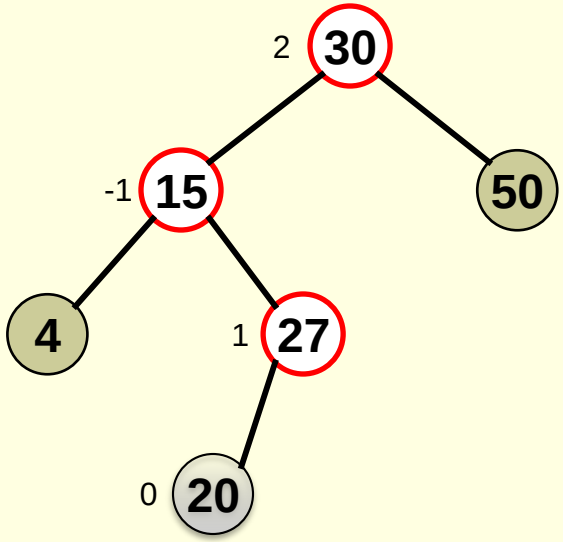


    - So now, follow the “Practical Rules” to rebalance this tree

    3. Then, take two steps, back down, towards the insertion point and mark those two nodes as well.

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Example 2:

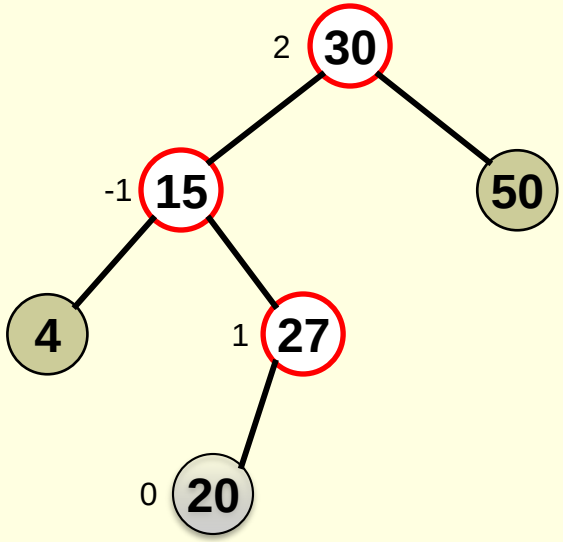


    - So now, follow the “Practical Rules” to rebalance this tree

    3. Then, take two steps, back down, towards the insertion point and mark those two nodes as well.

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Example 2:

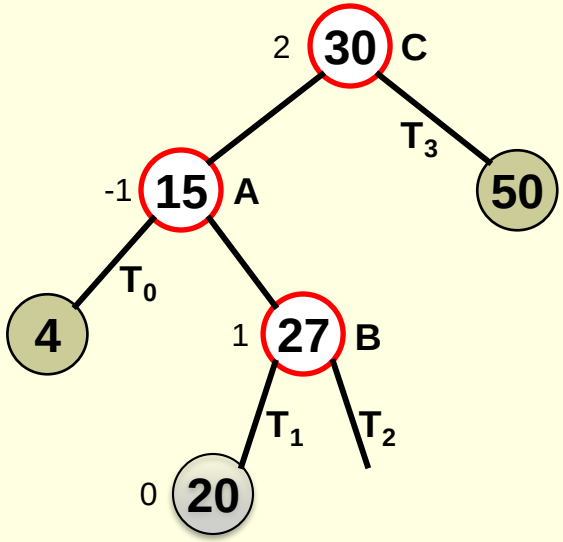


    - So now, follow the “Practical Rules” to rebalance this tree

    4. Label those ‘A, B, C’ nodes appropriately (and subtrees

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Example 2:

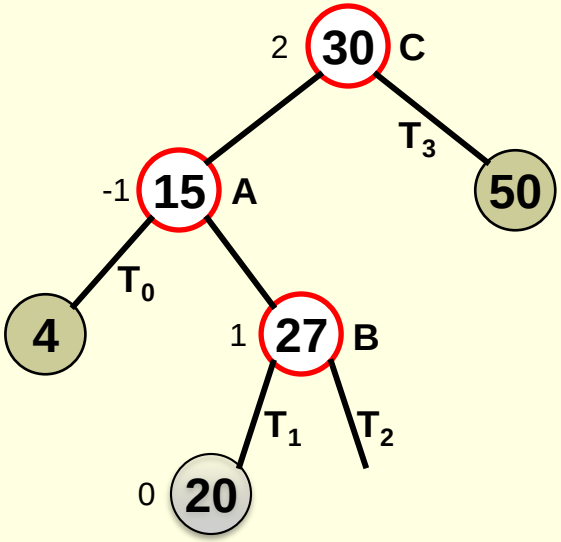


    - So now, follow the "Practical Rules" to rebalance this tree

4. Label those 'A, B, C' nodes appropriately (and subtrees)
   - Don't forget to label the subtrees from smallest to largest (from T_0 to T_3

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Example 2:



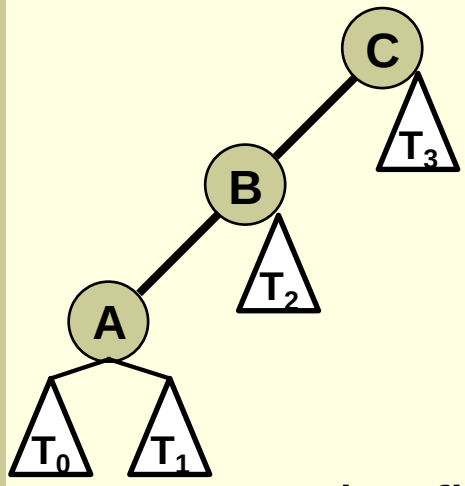
    - So now, follow the “Practical Rules” to rebalance this tree

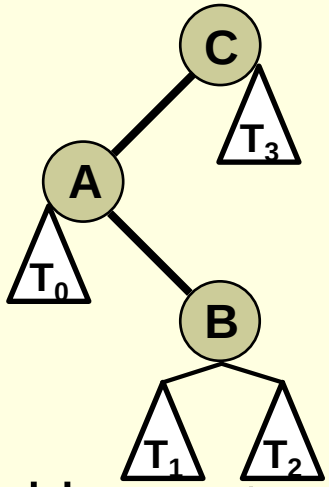
    5. Restructure those three nodes (and their subtrees)

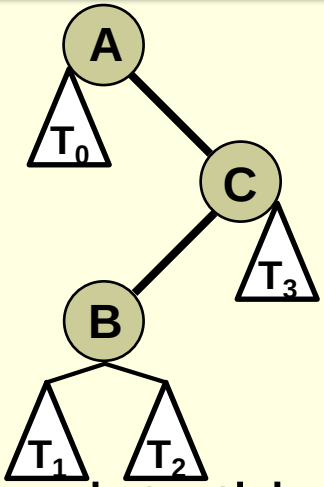
AVL Trees: Insertion Revisited

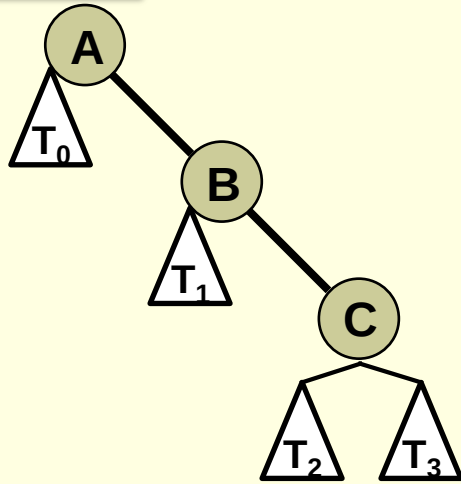
- AVL Trees: Insertion

All 4 of these trees:

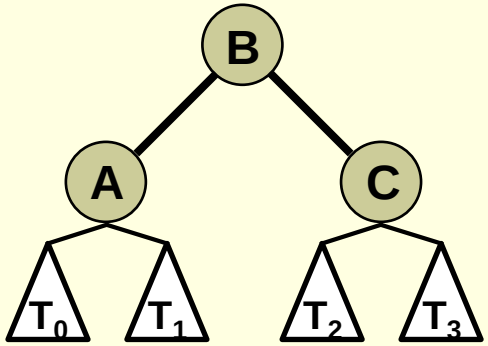






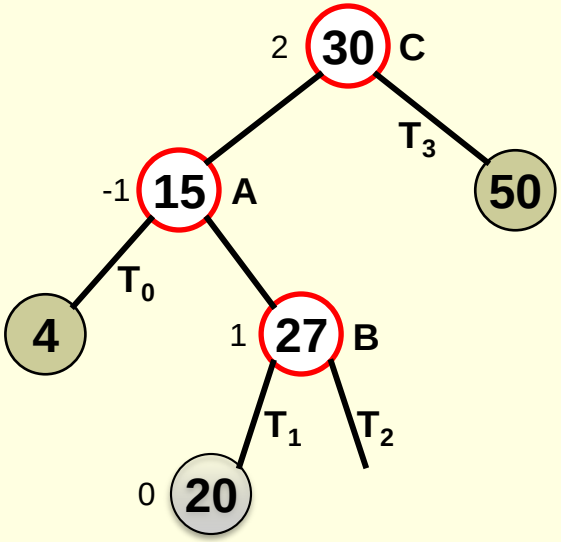


- Can be fixed by restructuring into this:



AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Example 2:

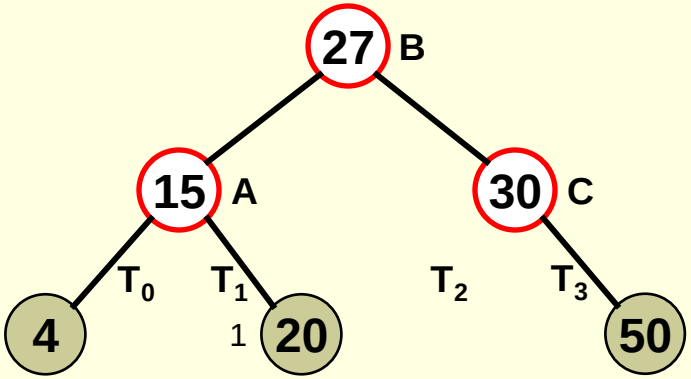


    - So now, follow the “Practical Rules” to rebalance this tree

    5. <u>Restructure</u> those three nodes (and their subtrees)

AVL Trees: Insertion Revisited

- AVL Trees: Insertion
  - Example 2:



- So now, follow the “Practical Rules” to rebalance this tree

5. Restructure those three nodes (and their subtrees)

   - So we simply restructure the tree according to the previous slide

Brief Interlude: FAIL Picture

AVL Trees: Insertion Revisited

- See PDF of Arup's Insertion notes
  - maybe not the most exciting notes
  -

AVL Trees: Insertion Revisited

WASN'T THAT MOMENTOUS!

AVL Trees: Insertion Revisited

Computer Science Department
University of Central Florida

COP 3502 – Computer Science I