DSA - Tutorial 10 (AVL Trees)

08-04-2025

1 Introduction

This document explains the balancing of AVL trees using the *right–left* convention, where the balance factor (BF) of a node is defined as:

$$\text{BF} = \text{height(right subtree)} - \text{height(left subtree)}$$

A node is balanced if its BF is -1, 0, or $+1$. The discussion covers the rotations required during insertion and deletion operations with detailed, step-by-step examples.

2 Rotation Cases for Insertion

Under the right–left convention, the following cases are considered when rebalancing after an insertion:

- **Single Left Rotation (RR Case):**
  When a node's BF becomes $+2$ (right heavy) and its right child's BF is ≥ 0. This usually happens when a new node is inserted into the right subtree of the right child.
- **Double Rotation – Right–Left (RL Case):**
  When a node's BF becomes $+2$ but its right child's BF is negative (i.e., the right child is left heavy). In this case, first perform a right rotation on the right child, then a left rotation on the node.
- **Single Right Rotation (LL Case):**
  When a node's BF becomes -2 (left heavy) and its left child's BF is ≤ 0. This occurs when a new node is inserted into the left subtree of the left child.
- **Double Rotation – Left–Right (LR Case):**
  When a node's BF becomes -2 but its left child's BF is positive (i.e., the left child is right heavy). First perform a left rotation on the left child, then a right rotation on the node.

3 Step-by-Step Insertion Example

Consider the insertion of the following elements in order: $10, 20, 30, 40, 50, 25$. Throughout, BF is computed as:

$$\text{BF} = \text{height(right)} - \text{height(left)}$$

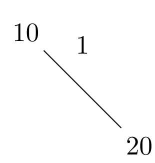
Step 1: Insert 10

10

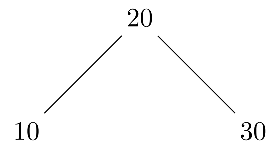
The tree consists of a single node, with $\text{BF}(10) = 0$.

Step 2: Insert 20

Since $20 > 10$, it becomes the right child of 10.



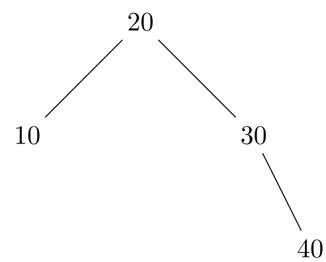
Rotation: A **Single Left Rotation** is performed at node 10. The resulting tree is:



Now, BF(10), BF(20), and BF(30) are 0.

Step 4: Insert 40

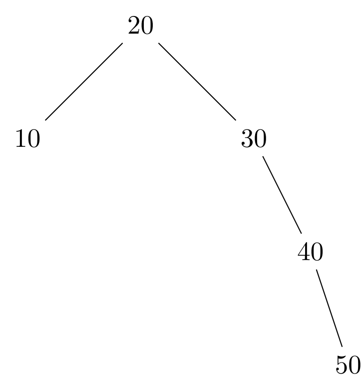
$40 > 20$ and $40 > 30$; it becomes the right child of 30.



BF(30) is now 1 and BF(20) becomes 1 (right subtree height 2, left subtree height 1).

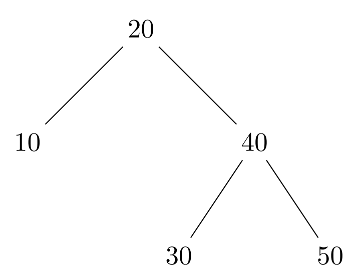
Step 5: Insert 50

$50 > 20$, $50 > 30$, and $50 > 40$; it becomes the right child of 40.



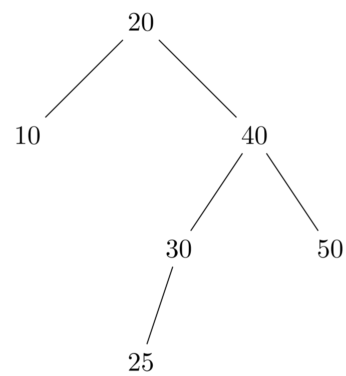
Here, BF(40) = 1 and BF(30) becomes 2 (with right subtree height 2, left height 0). This is an RR case.

Rotation: Perform a **Single Left Rotation** at node 30. The tree becomes:



Step 6: Insert 25

Since $25 > 20$ but $25 < 40$, compare with 30 (the left child of 40); as $25 < 30$, it becomes the left child of 30.



Recalculating the balance factors:

- BF(30) = (height(right) - height(left)) $= 0 - 1 = -1$.
- BF(40) = (left height 2 - right height 1) $= -1$.
- BF(20) = (right height 3 - left height 1) $= 2$.

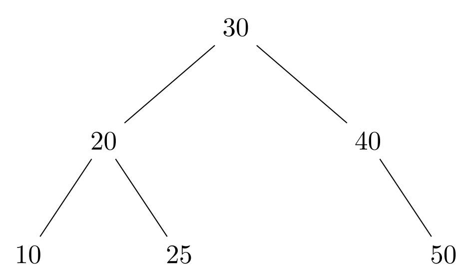
The imbalance at node 20 (BF $= 2$) along with its right child 40 being left heavy (BF $= -1$) indicates an RL case.

Rotation: A **Double Rotation** is needed at node 20:

(a) **Right Rotation** on node 40 makes node 30 the new root of the right subtree.

(b) **Left Rotation** on node 20 rebalances the tree.

The final tree after insertion is:



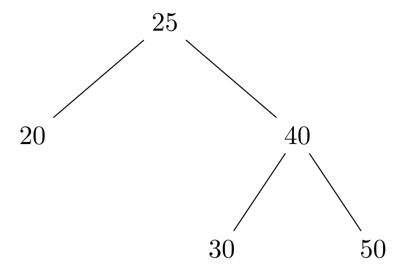
4 Rotation Cases for Deletion

After deletion, an AVL tree might become unbalanced. The rotation rules are analogous:

- If a node's BF becomes $+2$ (right heavy):
  - If its right child's BF is ≥ 0, perform a **Single Left Rotation**.
  - If its right child's BF is < 0, perform a **Double Rotation** (first a right rotation on the right child, then a left rotation on the node).
- If a node's BF becomes -2 (left heavy):
  - If its left child's BF is ≤ 0, perform a **Single Right Rotation**.
  - If its left child's BF is > 0, perform a **Double Rotation** (first a left rotation on the left child, then a right rotation on the node).

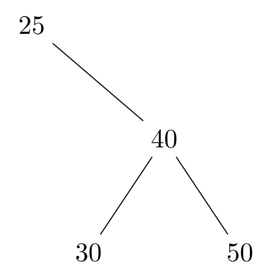
4.1 Deletion Example 1 (Single Rotation)

Consider the following starting tree with 5 nodes:



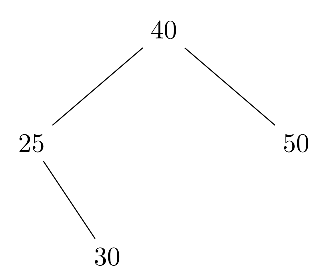
Operation: Delete node 20 (a leaf).

The resulting tree is:



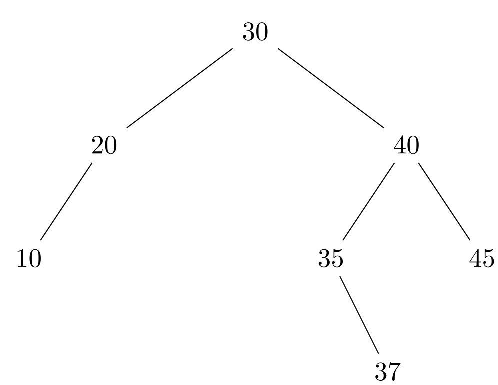
Here, BF(25) becomes 2 (right subtree height 2, left subtree height 0). Since the right child 40 has BF = 0, a **Single Left Rotation** is performed at node 25.

The final tree after rotation is:



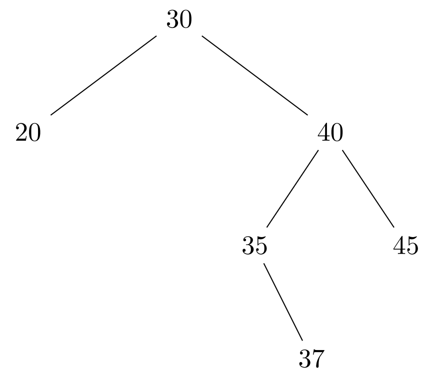
4.2 Deletion Example 2 (Double Rotation)

Consider a starting tree with 7 nodes:



Operation: Delete node 10 from the left subtree.

After deletion, the tree becomes:



Now, BF(30) becomes +2 (right heavy) due to the decreased height of its left subtree, and suppose node 40 has a BF of −1 (left heavy). This situation indicates an RL case.

Rotation: Perform a **Double Rotation**:

(a) First, perform a **Right Rotation** on node 40, making node 35 the new root of that subtree.

(b) Then, perform a **Left Rotation** on node 30.

The final tree after the double rotation becomes: