Height Measuring Formulae for Different Trees

Here's a summary of height calculation and the number of nodes formulas for Binary Trees, AVL Trees, Binary Search Trees (BST), and Heaps:

1. Binary Tree

In a binary tree, the maximum number of nodes at level n (where the root is considered level 0) is:

$$2^n$$

This is because, at each level nnn, the number of nodes can double compared to the previous level. Therefore:

- **Level 0**: $2^0=1$ node (the root).
- **Level 1**: $2^1=2$ nodes.
- **Level 2**: $2^2=4$ nodes.
- **Level n**: 2^n nodes.

The maximum number of nodes in a binary tree of height hhh (where height is the number of edges from the root to the farthest leaf) is:

2h+1−12^{h+1} - 12h+1−1

Explanation:

In a full binary tree:

- Level 0 (the root) has 20=12^0 = 120=1 node.
- Level 1 has $1=2^1$ = 221=2 nodes.
- Level 2 has $2=2^2$ = 422=4 nodes.
- Level h has 2^h nodes.

Summing all nodes from level 0 to level hhh:

$$1+2+4+\cdots+2^h=2^{h+1}-1$$

Thus, a binary tree of height h can have up to $2^{h+1} - 1$ nodes in total.

Alternative Formulation

If the binary tree has n nodes, the minimum height h it can have is:

$$h = \lfloor \log 2(n+1) \rfloor - 1$$

This height represents the balanced or complete form of the binary tree, where the tree is as compact as possible.

2. AVL Tree (Balanced Binary Search Tree)

An AVL Tree is a self-balancing binary search tree that maintains a balance factor of −1,0, or +1 at each node to prevent skewing.

Height Calculation

For an AVL tree:

- **Minimum number of nodes required to achieve a height h** is given by the recursive formula:
  $$N(h) = N(h-1) + N(h-2) + 1$$
  where N(0)=1

  and N(1)=2

  Maximum height h for n nodes:
  $$h \approx 1.44 \log 2(n+2) - 0.328$$

h \approx 1.44 \log_2(n + 2) - 0.328

which ensures it is balanced.

Number of Nodes

- **Minimum nodes for a given height h**: Follows the sequence above, ensuring the balance constraint.

3. Binary Search Tree (BST)

A Binary Search Tree is a binary tree with the property that each node's left subtree contains values less than the node's value, and the right subtree contains values greater than the node's value.

Height Calculation

- **Best case (balanced BST)**: **Height $h = \log 2(n+1) - 1$**
- **Worst case (skewed BST)**: **Height $h = n - 1$** (where each node has only one child).

Number of Nodes

- **Minimum nodes at height h: $h+1$** (for a completely skewed tree).
- **Maximum nodes at height h: $2^{\{h+1\}} - 1$** (if the tree is complete).

4. Heap

In a binary heap, each level (except possibly the last) is fully filled, and elements follow a specific order property (either min-heap or max-heap).

Height Calculation

For a complete binary heap:

- **Height h for n nodes**: **$h = \log 2(n)$**
- since a complete binary heap is always as compact as possible.

Number of Nodes

- **Maximum nodes at height h: $2^{\{h+1\}} - 1$.**

- **Minimum nodes at height** $\mathbf{h}$**:** $\mathbf{h=2^h}$ (minimum nodes at the last level).