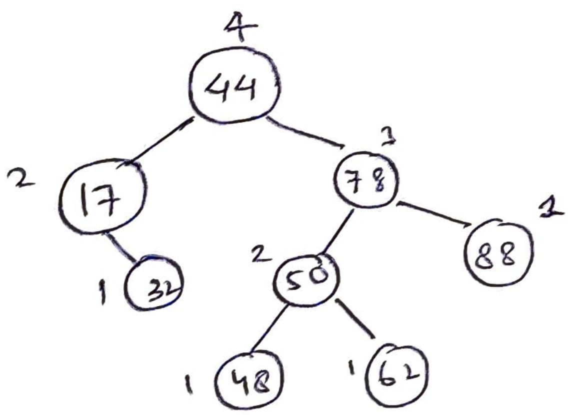
DSA IIT Delhi - AVL Trees (Lec - 11)

Binary search tree → insertion, deletion, take O(height) time.

→ AVL trees are balanced

→ An AVL tree is a BST such that for every internal node v of T, the heights of the children of v can differ by atmost 1.

ex.



height of a node is height of subtree rooted at that node + 1

Height of an AVL Tree

→ The height of an AVL tree storing n keys is O(log n).

→ Justification → Find n(h): the minimum number of nodes in an AVL tree of height h.

$n(1) = 1$, $n(2) = 2$

→ For $h \geq 3$, an AVL tree of height h contains root node, one AVL subtree of height (h-1) and other AVL subtree of height (h-1) or (h-2).

i.e $n(h) = 1 + n(h-1) + n(h-2)$

$n(3) = 4$

→ knowing $n(h-1) \geq n(h-2)$, we get

$$n(h) = n(h-1) + n(h-2) + 1 > 2n(h-2)$$

$$\begin{aligned} n(h) &> 2n(h-2) \\ &> 4n(h-4) \\ &> 8n(h-6) \\ &\vdots \\ &> 2^i n(h-2i) \end{aligned}$$

when $i = h/2 - 1$, we get $n(h) > 2^{n/2 - 1} \cdot n(2)$
$= 2^{h/2}$

$$\left[n(h) > 2^{h/2} \right].$$

→ Thus, if AVL tree has height h, then it atleast has $2^{h/2}$ nodes.

⇒ $h < 2 \log n(h)$.

⇒ Thus, if we have an AVL tree of n node then, $h < 2 \log n$.

A sharper bound on height of AVL tree.

→ We prove using induction that the minimum number of nodes in an AVL tree of height h, $n(h) >= c^h$, where c is some number > 1.

→ Base case $h = 1$. Now $n(h) \geq c > 1$

→ $n(k) = c^{k-1} + c^{k-2}$

$n(k) \geq c^k$ if $c^{k-1} + c^{k-2} \geq c^k$

$c^2 - c - 1 \leq 0$ $\quad c \in \left[\frac{1-\sqrt{5}}{2}, \frac{1+\sqrt{5}}{2}\right]$

highest $c = 1.63$.

→ Hence, AVL tree on n nodes has height atmost $\log_{1.63} n$

→ h, n.

$n \geq n(h) \geq (1.63)^{h-1}$

$\Rightarrow \log_{1.63} n = (h-1)$

$\Rightarrow h = \log_{1.63} n + 1$.

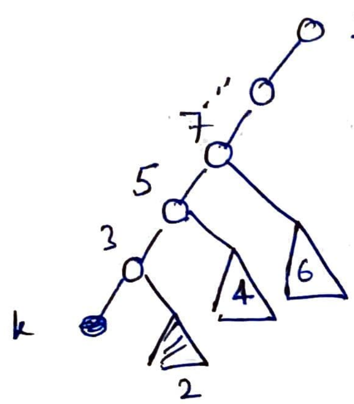
structure of an AVL tree

→ Consider an AVL tree on n nodes.

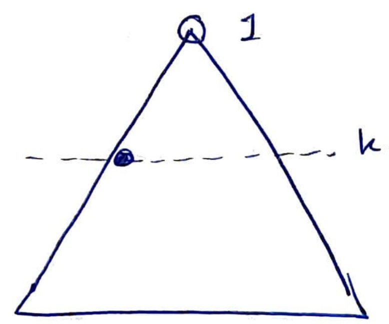
→ Consider a leaf closest to the ~~nodes~~ root.

→ Suppose it is at level k.

→ We will show that height of tree is atmost $(2k-1)$.



max height
$=(2k-1)$



→ Claim: Since closest leaf has level k, all nodes at level $1, 2, \dots (k-2)$ have 2 children.

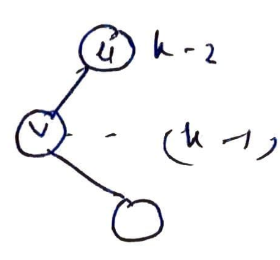
Proof by contradiction :-

→ Suppose node u at level $k-2$ has only 1 child v.

→ v is at level $(k-1)$ and so cannot be a leaf.

→ Hence, subtree rooted at v has height atleast 2.

→ Height balance property is violated at u.



→ By previous claim, all levels 1 to $(k-1)$ are full

→ Hence, tree has atleast 2^{k-1} nodes

→ Since height of tree ~~has~~ is atmost $(2k-1)$, it has at most 2^{2k-1} nodes

→ Thus, $2^{k-1} \le n \le 2^{2k-1}$

→ Substituting h.

$$2^{\frac{(h-1)}{2}} \le n \le 2^{h}.$$

Summary

(i) Height is h, then leaf which is closest to the root is at level atleast $(h+1)/2$.

(ii) On the first $(h-1)/2$ levels, the AVL tree is a complete binary tree.

(iii) No of nodes is at least $2^{(h-1)/2}$ and atmost 2^{h}.

Insertion

→ A binary tree is called height balanced if for every node v, height of v's children differ by atmost 1.

→ Inserting a node into an AVL tree changes the heights of some nodes of tree.

→ If insertion causes T to become unbalanced, we travel up the tree from the newly created node until we find the first node x such that its grandparent z is unbalanced node.

→ Let y be parent of node x.

→ To rebalance the subtree rooted at z, we must perform a rotation.