Topic 7.6

Extra slides: AVL trees (In GATE/GRE syllabus)

AVL (Adelson, Velsky, and Landis) tree

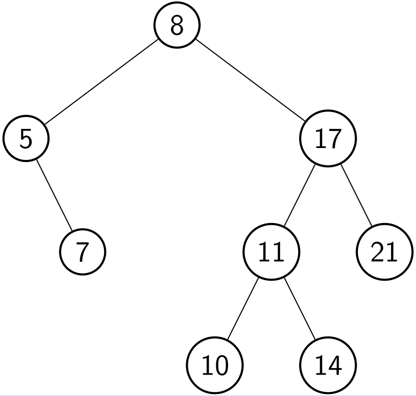
Definition 7.3

An AVL tree is a binary search tree such that for each node n

$$|height(right(n)) - height(left(n))| \leq 1.$$

Example 7.8

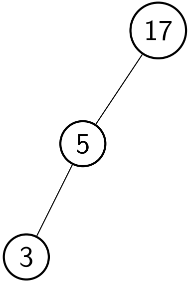
An example of an AVL tree.

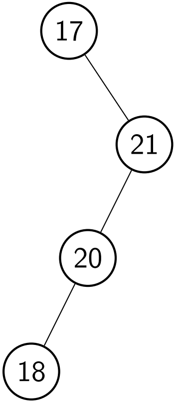


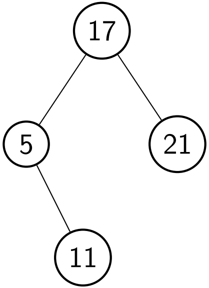
Exercise: Identify the AVL trees

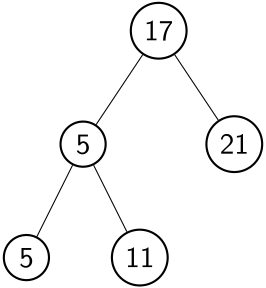
Exercise 7.15

Which of the following are AVL trees?









Topic 7.7

Height of AVL tree

AVL tree height

Theorem 7.1

The height of an AVL tree T having n nodes is $O(\log n)$.

Proof.

Let $n(h)$ be the minimum number of nodes for height h.

Base case:

$n(1) = 1$ and $n(2) = 2$.

Induction step:

Consider an AVL tree with height $h \geq 3$. In the minimum case, one child will have a height of $h - 1$ and the other child will have a height of $h - 2$. (Why?)

Therefore, $n(h) = 1 + n(h-1) + n(h-2)$. ...

Commentary: We need to show that $n(h) > n(h-1)$ is monotonous. Ideally, $n(h) = 1 + n(h-1) + min(n(h-2), n(h-1))$. This proves that $n(h) > n(h-1)$.

AVL tree height(2)

Proof(continued.)

Since $n(h-1) > n(h-2)$,

$$n(h) > 2n(h-2).$$

Therefore,

$$n(h) > 2^i n(h-2i).$$

For $i = h/2 - 1$(Why?),

$$n(h) > 2^{h/2-1} n(2) = 2^{h/2}.$$

Commentary: Here is the explanation of the last step. Consider an AVL tree with m nodes and h height. By definition, $h(n) \leq m$. Since $h < 2\log n(h)$, $h < 2\log m$. Therefore, h is $O(\log m)$.

Therefore,

$$h < 2\log n(h).$$

Therefore, the height of an AVL tree is $O(\log n)$. (Why?)

Closest leaf

Theorem 7.2

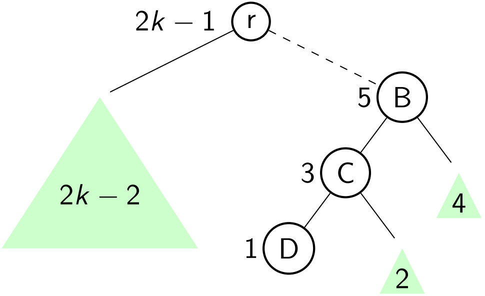
Let T be an AVL tree. Let the level of the closest leaf to the root of T is k.

$$height(T) \leq 2k - 1$$

Proof.

Let D be the closest leaf of the tree.

- The height of $right(C)$ cannot be more than 2. (Why?)
- Therefore, the maximum height of C is 3.
- Therefore, the maximum height of $right(B)$ is 4.
- Therefore, the maximum height of B is 5.
- Continuing the argument, the maximum height of root r is $2k - 1$.



□

A part of AVL is a complete tree

Theorem 7.3

Let T be an AVL tree. Let the level of the closest leaf to the root of T is k. Upto level $k-2$ all nodes have two children.

Proof.

A node at level $k-2-i$ cannot be a leaf. (Why?)

Let us assume that a node n at level $k-2-i$ has a single child n'.

The height of n' cannot be more than 1. (Why?)

Therefore, n' is a leaf. Contradiction. $\square$

Exercise 7.16

Show T has at least 2^{k-1} nodes.

Another proof of tree height bound

Let T have n nodes and the height of T be h.

We know the following from the previous theorems.

- $n \geq 2^{k-1}$, and
- $2k - 1 \geq h$.

Therefore,

$$n \geq 2^{k-1} \geq 2^{(h-1)/2}$$

Exercise 7.17

What is the maximum number of nodes given height h?

Problem: A sharper bound for the AVL tree

Exercise 7.18

a. *Find largest* c *such that* $c^{k-2} + c^{k-1} \geq c^k$

b. *Recall* $n(h) = 1 + n(h-1) + n(h-2)$*. Let* c_0 *be the largest* c*. Show that* $n(h) \geq c^{h-1}$*.*

c. *Prove that the above bound is a sharper bound than our earlier proof.*

Topic 7.8

Insertion and deletion in AVL trees

Insert and delete

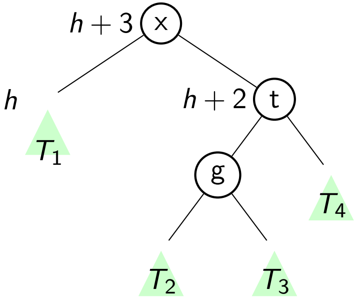
Insert and delete like BST.

At most a path to one node may have height imbalances of 2. (Why?)

We have to repair height imbalances by rotations around the deepest imbalanced node.

Rebalancing AVL trees

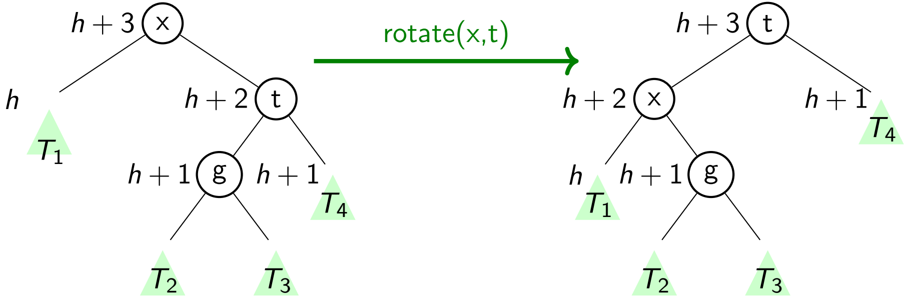
Let x be the deepest imbalanced node. Let t be the taller child. Let g be the grandchild via t that is not on the straight path from x.



Three cases:

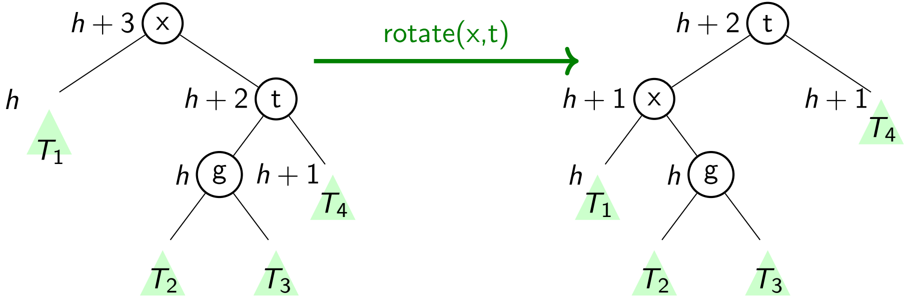
1. Case 1: Height of g is $h+1$ and T_4 is $h+1$.
2. Case 2: Height of g is h and T_4 is $h+1$.
3. Case 3: Height of g is $h+1$ and T_4 is h.

Case 1: Both grandchildren via t have height $h+1$



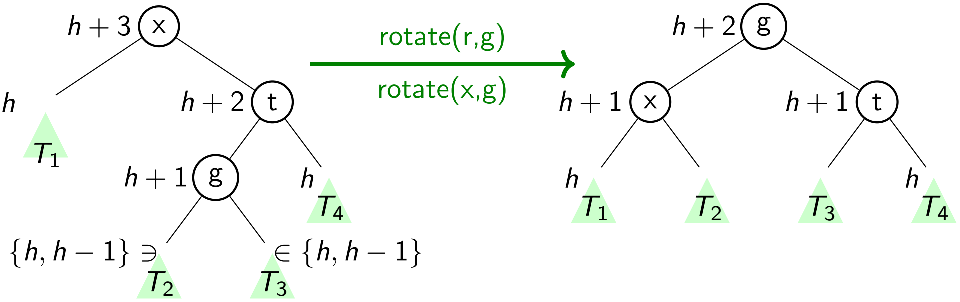
The imbalance in the subtree is repaired. We check the parent of t.

Case 2: Right-left grandchild has height h



Imbalance is repaired. But, the parent of t may need repair.

Case 3: Right right grandchild has height h



Imbalance is repaired. But, the parent may need repair.

Complexity of insertion/deletion

Exercise 7.19

a. What is the bound on the number of rotations for a single insert/delete?

b. Compare the bounds with RB trees insertion/deletion.

c. Which definition is more strict RB or AVL? Or, are they incomparable?