



CS 225

Data Structures

March 6 – AVL Analysis

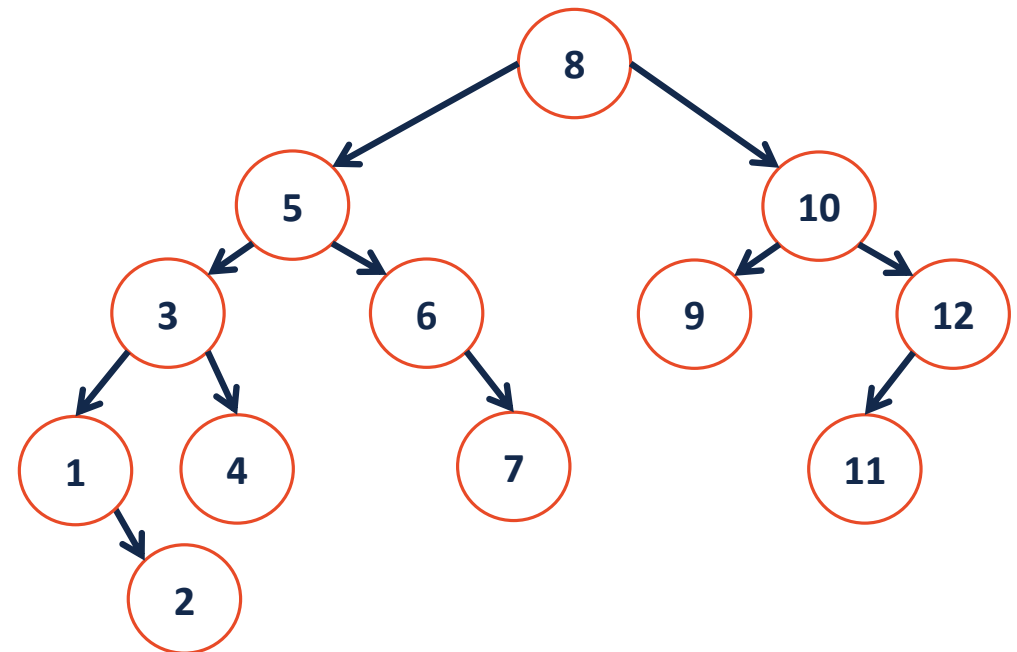
G Carl Evans

Insertion into an AVL Tree

`_insert(6.5)`

Insert (pseudo code):

- 1: Insert at proper place
- 2: Check for imbalance
- 3: Rotate, if necessary
- 4: Update height

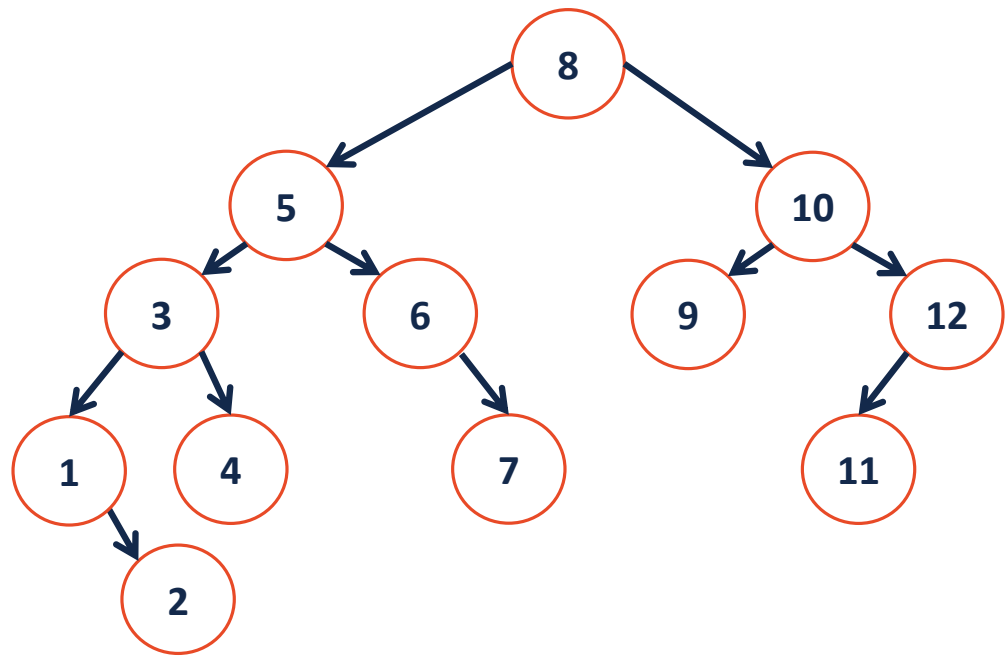


```
1 struct TreeNode {
2     T key;
3     unsigned height;
4     TreeNode *left;
5     TreeNode *right;
6 };
```

```

1  template <class T> void AVLTree<T>::_insert(const T & x, treeNode<T> * & t ) {
2      if( t == NULL ) {
3          t = new treeNode<T>( x, 0, NULL, NULL);
4      }
5
6      else if( x < t->key ) {
7          _insert( x, t->left );
8          int balance = height(t->right) - height(t->left);
9          int leftBalance = height(t->left->right) - height(t->left->left);
10         if ( balance == -2 ) {
11             if ( leftBalance == -1 ) { rotate_____ ( t ); }
12             else { rotate_____ ( t ); }
13         }
14     }
15
16     else if( x > t->key ) {
17         _insert( x, t->right );
18         int balance = height(t->right) - height(t->left);
19         int rightBalance = height(t->right->right) - height(t->right->left);
20         if( balance == 2 ) {
21             if( rightBalance == 1 ) { rotate_____ ( t ); }
22             else { rotate_____ ( t ); }
23         }
24     }
25
26     t->height = 1 + max(height(t->left), height(t->right));
27 }

```





AVL Tree Analysis

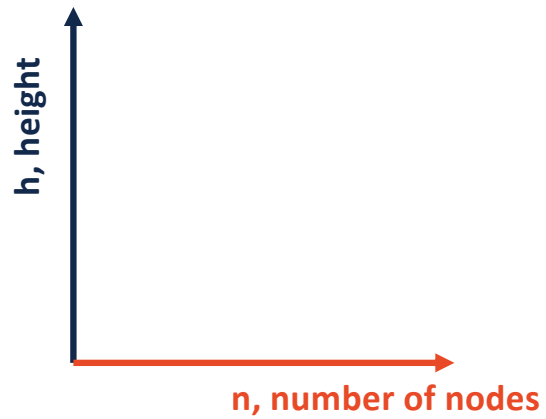
We know: insert, remove and find runs in: _____.

We will argue that: h is _____.

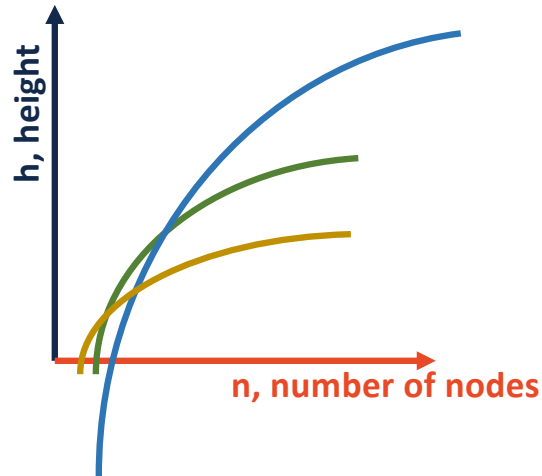
AVL Tree Analysis

Definition of big-O:

...or, with pictures:

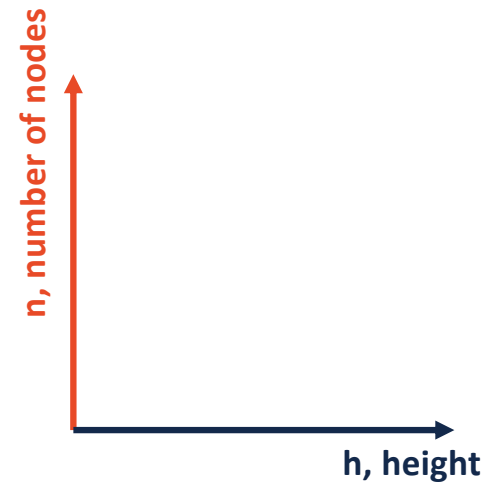
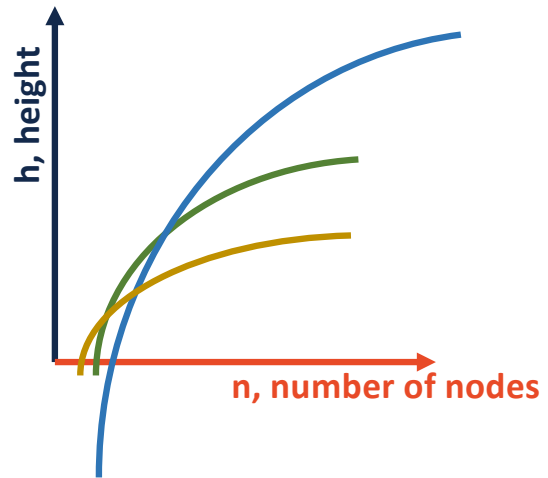


AVL Tree Analysis

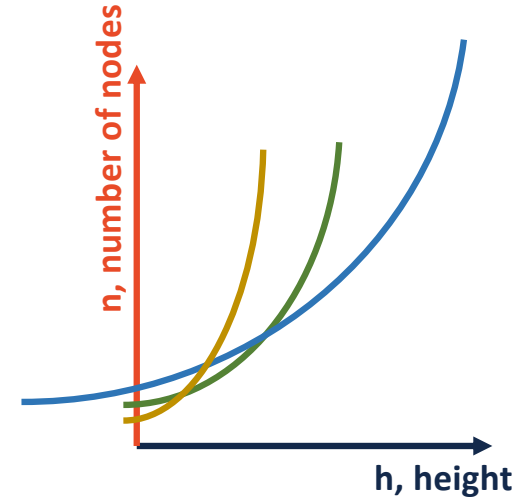
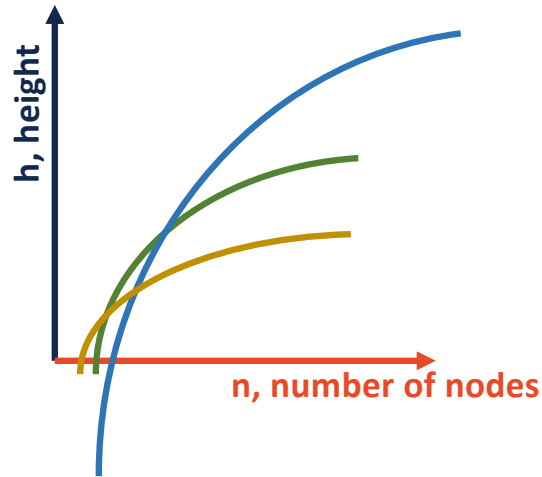


- The height of the tree, $f(n)$, will always be less than $c \times g(n)$ for all values where $n > k$.

AVL Tree Analysis



AVL Tree Analysis



- The number of nodes in the tree, $f^{-1}(h)$, will always be greater than $c \times g^{-1}(h)$ for all values where $n > k$.



Plan of Action

Since our goal is to find the lower bound on n given h , we can begin by defining a function given h which describes the smallest number of nodes in an AVL tree of height h :



Simplify the Recurrence

$$N(h) = 1 + N(h - 1) + N(h - 2)$$

State a Theorem

Theorem: An AVL tree of height h has at least _____.

Proof:

I. Consider an AVL tree and let h denote its height.

II. Case: _____

An AVL tree of height _____ has at least _____ nodes.



Prove a Theorem

III. Case: _____

An AVL tree of height _____ has at least _____ nodes.



Prove a Theorem

IV. Case: _____

By an Inductive Hypothesis (IH):

We will show that:

An AVL tree of height _____ has at least _____ nodes.



Prove a Theorem

V. Using a proof by induction, we have shown that:

...and inverting:



Summary of Balanced BST

Red-Black Trees

- Max height: $2 * \lg(n)$
- Constant number of rotations on insert, remove, and find

AVL Trees

- Max height: $1.44 * \lg(n)$
- Rotations:



Summary of Balanced BST

Pros:

- Running Time:
 - Improvement Over:
- Great for specific applications:



Summary of Balanced BST

Cons:

- Running Time:

- In-memory Requirement: