Binary Search Trees: AVL Trees

Daniel Kane

Department of Computer Science and Engineering
University of California, San Diego

Data Structures
Data Structures and Algorithms

Learning Objectives

- Understand what the height of a node is.
- State the AVL property.
- Show that trees satisfying the AVL property have low depth.

Outline

1. Basic Idea

2. Analysis

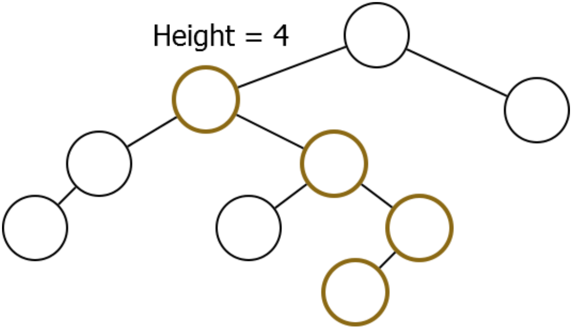
Balance

- Want to maintain balance.
- Need a way to measure balance.

Height

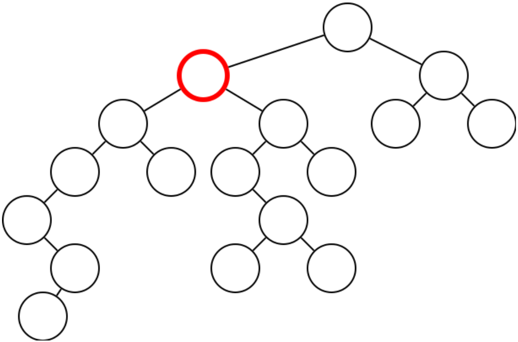
Definition

The height of a node is the maximum depth of its subtree.



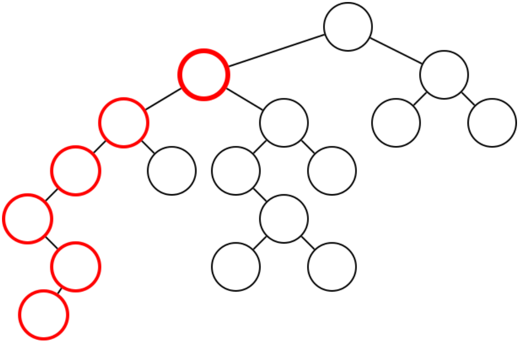
Problem

What is the height of the selected node?



Problem

What is the height of the selected node?



Recursive Definition

$N.\texttt{Height}$ equals

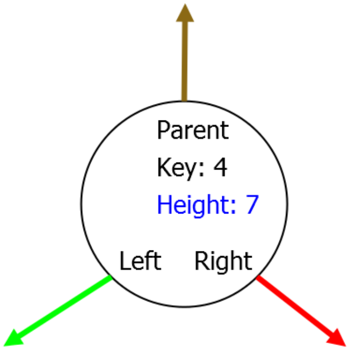
1 if N is a leaf,

$$1 + \max(N.\texttt{Left.Height}, N.\texttt{Right.Height})$$

otherwise.

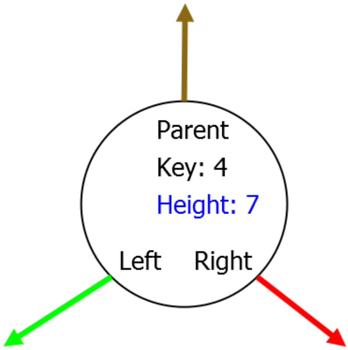
Field

Add height field to nodes.



Field

Add height field to nodes.



(Note: We'll have to work to ensure that this is kept up to date)

Balance

- Height is a rough measure of subtree size.
- Want size of subtrees roughly the same.
- Force heights to be roughly the same.

AVL Property

AVL trees maintain the following property:
For all nodes N,

$$|N.\mathtt{Left}.\mathtt{Height} - N.\mathtt{Right}.\mathtt{Height}| \leq 1$$

We claim that this ensures balance.

Outline

1. Basic Idea
2. Analysis

Idea

Need to show that AVL property implies
$\text{Height} = O(\log(n))$.

Idea

Need to show that AVL property implies $\mathrm{Height} = O(\log(n))$.

Alternatively, show that large height implies many nodes.

Result

Theorem

Let N be a node of a binary tree satisfying the AVL property. Let $h = N.\texttt{Height}$. Then the subtree of N has size at least the Fibonacci Number F_h.

Recall

$$F_n = \begin{cases} 0, & n = 0\,, \\ 1, & n = 1\,, \\ F_{n-1} + F_{n-2}, & n > 1\,. \end{cases}$$

Recall

$$F_n = \begin{cases} 0, & n = 0\,, \\ 1, & n = 1\,, \\ F_{n-1} + F_{n-2}, & n > 1\,. \end{cases}$$

$F_n \geq 2^{n/2}$ for $n \geq 6\,.$

Proof

Proof.

By induction on h.

Proof

Proof.

By induction on h.

If $h = 1$, have one node.

Proof

Proof.

By induction on h.
If $h = 1$, have one node.
Otherwise, have one subtree of height $h - 1$ and another of height at least $h - 2$.
By inductive hypothesis, total number of nodes is at least $F_{h-1} + F_{h-2} = F_h$. $\square$

Large Subtrees

So node of height h has subtree of size at least $2^{h/2}$.
In other words, if n nodes in the tree, have height $h \leq 2\log_2(n) = O(\log(n))$.

Conclusion

AVL Property

If you can maintain the AVL property, you can perform operations in $O(\log(n))$ time.