March 4 – AVL Analysis

G Carl Evans
AVL Tree Analysis

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

We will argue that: h is ____________.
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

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

AVL Trees
- Max height: $1.44 \times \lg(n)$
- Rotations:
Summary of Balanced BST

Pros:
- Running Time:
  - Improvement Over:

- Great for specific applications:
Summary of Balanced BST

**Cons:**
- Running Time:

- In-memory Requirement:
Range-based Searches

Q: Consider points in 1D: \( p = \{p_1, p_2, \ldots, p_n\} \).

...what points fall in [11, 42]?

Tree construction:
Range-based Searches

Balanced BSTs are useful structures for range-based and nearest-neighbor searches.

Q: Consider points in 1D: \( p = \{p_1, p_2, \ldots, p_n\} \).
...what points fall in [11, 42]?

Ex: 3 6 11 33 41 44 55
Range-based Searches

Q: Consider points in 1D: $p = \{p_1, p_2, \ldots, p_n\}$.

...what points fall in $[11, 42]$?

Ex:

3 6 11 33 41 44 55
Range-based Searches

Q: Consider points in 1D: \( p = \{p_1, p_2, \ldots, p_n\} \).
   ...what points fall in \([11, 42]\)?

Tree construction:
Range-based Searches
Range-based Searches
Q: Consider points in 1D: $p = \{p_1, p_2, \ldots, p_n\}$. ...what points fall in $[11, 42]$?
Range-based Searches
Running Time
Range-based Searches

Q: Consider points in 1D: \( p = \{p_1, p_2, \ldots, p_n\} \).

...what points fall in \([11, 42]\)?

Ex: 3 6 11 33 41 44 55
Red-Black Trees in C++

C++ provides us a balanced BST as part of the standard library:

```cpp
std::map<K, V>
```

```cpp
V & std::map<K, V>::operator[](const K &)
```

```cpp
iterator std::map<K, V>::lower_bound( const K & )
iterator std::map<K, V>::upper_bound( const K & )
```
<table>
<thead>
<tr>
<th></th>
<th>Unsorted Array</th>
<th>Sorted Array</th>
<th>Unsorted List</th>
<th>Sorted List</th>
<th>Binary Tree</th>
<th>BST</th>
<th>AVL</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Find</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Insert</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Remove</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Traverse</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
CS 225 Final Project

Working with data and using graphs
The Internet 2003
The OPTE Project (2003)
Map of the entire internet; nodes are routers; edges are connections.
Conflict-Free Final Exam Scheduling Graph

Unknown Source

Presented by Cinda Heeren, 2016
“Rush Hour” Solution
Unknown Source
Presented by Cinda Heeren, 2016
Class Hierarchy At University of Illinois Urbana-Champaign
A. Mori, W. Fagen-Ulmschneider, C. Heeren

Graph of every course at UIUC; nodes are courses, edges are prerequisites

http://waf.cs.illinois.edu/discovery/class_hierarchy_at_illinois/
B-Tree Motivation

In Big-O we have assumed uniform time for all operations, but this isn’t always true.

However, seeking data from the cloud may take 40ms+. ...an O(lg(n)) AVL tree no longer looks great:
BTree Motivations

Knowing that we have large seek times for data, we want to: