AVL Insertion

AVL Removal

Running Times:

<table>
<thead>
<tr>
<th></th>
<th>AVL Tree</th>
</tr>
</thead>
<tbody>
<tr>
<td>find</td>
<td></td>
</tr>
<tr>
<td>insert</td>
<td></td>
</tr>
<tr>
<td>remove</td>
<td></td>
</tr>
</tbody>
</table>

Motivation:
Big-O is defined as:

Visually:

\[ f(n), g(n) \]

- The graph above describes functions of the height \( h \) of an AVL tree given the number of nodes \( n \).
- Inverse functions describe the number of nodes in a tree \( n \) given a height \( h \).

Plan of Action:
Goal: Find a function that defines the lower bound on \( n \) given \( h \).

Given the goal, we begin by defining a function that describes the smallest number of nodes in an AVL of height \( h \):
State a Theorem:
An AVL tree of height $h$ has at least ________________.

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

II. Case: ________________

III. Case: ________________

IV. Case: ________________

By an inductive hypothesis (IH):

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

...and by inverting our finding:

<table>
<thead>
<tr>
<th>Summary of Balanced BSTs:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Advantages</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Iterators + Usefulness</th>
</tr>
</thead>
</table>

Three weeks ago, you saw that you can use an iterator to loop through data:

```
DFS dfs(...);
for ( ImageTraversal::Iterator it = dfs.begin(); it != dfs.end(); ++it ) {
    std::cout << (*it) << std::endl;
}
```

You will use iterators extensively in MP4, creating them in Part 1 and then utilizing them in Part 2. Given the iterator, you can use the for-each syntax available to you in C++:

```
DFS dfs(...);
for ( const Point & p : dfs ) {
    std::cout << p << std::endl;
}
```

The exact code you might use will have a generic `ImageTraversal`:

```
ImageTraversal & traversal = /* ... */;
for ( const Point & p : traversal ) {
    std::cout << p << std::endl;
}
```

CS 225 – Things To Be Doing:

1. Theory Exam 2 is ongoing!
2. MP4 extra credit submission ongoing – due Monday, March 5th!
3. lab_huffman is due on Sunday, March 4th
4. Daily POTDs are ongoing!