# Tree Analysis

Matthew Galitz

University of Virginia, Charlottesville, VA 22904

**Abstract.**
The purpose of the experiment is to compare the efficiency of an AVL Tree (referred to throughout as a AVLT), versus a Binary Search Tree (referred to throughout as a BST)

## 1 Introduction

### 1.1 Problem

The problem is to determine the relative efficiencies of the methods of the AVLT and the BST.

### 1.2 Description

A BST is a data structure consisting of nodes, each containing a piece of data. It begins with one root node. The root node points to a left and right node with the same properties as the root node. The nodes at the bottom of the tree are called the leaves. A AVLT is similar to a BST except every node must be balanced, such that the longest path from left node to leaf and the longest path from right node to leaf must have a difference of at most one node. The AVLT rearranges the orientation of the nodes to maintain this balance, unlike a BST.

## 2 Methods

First each method was executed 150 times on a BST & AVLT. These nodes inserted contain random common words of varying length. Again, each method was executed 150 times on a BST & AVLT. The nodes inserted contain random integers from 1 to 10,000. The amount of time it takes to complete 150 executions of each method is recorded in milliseconds for five trials.

**Number of Executions for each Group**

| Methods | Random Data Set | Random Integers |
|---|---|---|
| find() | 1000 | 1000 |
| insert() | 1000 | 1000 |

The average and standard deviation of the data for each method are calculated using the following equations respectively.

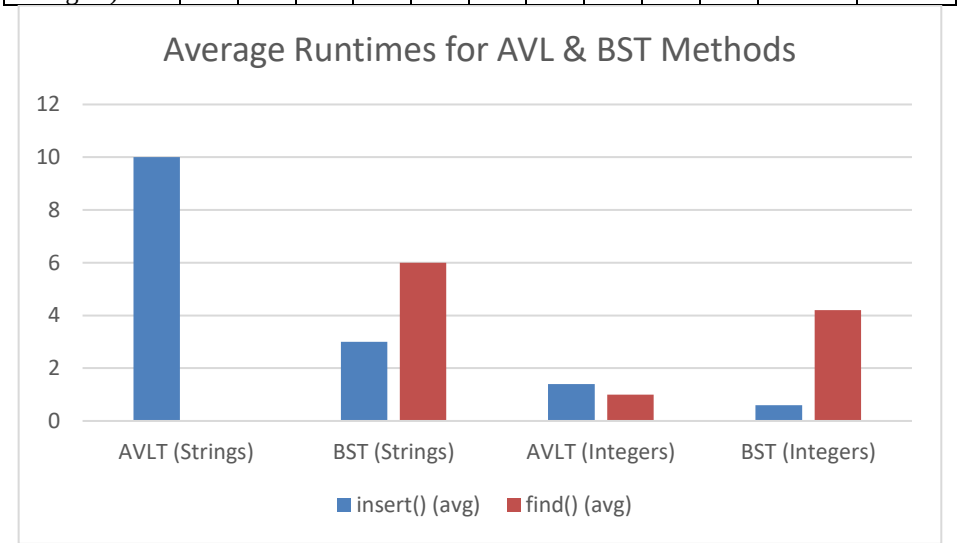
$$A = \frac{\sum_{n=1}^{N} x_n}{N}$$

Where $x_n$ is the time elapsed for trial $n$, $N$ is the total number of trials, and $A$ is the average time for a particular method.

## 3 Results

**Runtime of each Method for 1,000 Nodes**

| | insert() (ms) | | | | | find() (ms) | | | | | Average | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| AVLT (Strings) | 10 | | | | | 0 | | | | | 10 | 0 |
| BST (Strings) | 3 | | | | | 6 | | | | | 3 | 6 |
| AVLT (Integers) | 2 | 1 | 1 | 1 | 2 | 1 | 0 | 1 | 1 | 2 | 1.4 | 1 |
| BST (Integers) | 1 | 1 | 0 | 0 | 1 | 3 | 5 | 4 | 5 | 4 | 0.6 | 4.2 |



## 4 Conclusion

The data indicates that the AVLT and BST are more efficient depending on the method. The BST is quicker to insert a node than the AVLT. This makes sense; an AVLT must verify if the tree is unbalanced and if so, balance the tree. A BST does not have to do this. The AVLT is quicker to find a node than the BST. This makes sense; an AVLT maintains a balanced structure and therefore prevents a long string of nodes from forming while a BST does not. The height of the BST is often larger than the AVLT. This makes sense because an AVLT maintains a balanced structure.