Comp605-C3 Dataset Analysis

Comp605 Assignment 3

ID: 21526940 C3-ChatGPT Conversation Prompts

Insert Files into a Binary Search Tree and AVL Tree				
File Name	Sequential Files		Random Files	
	BST	AVL	BST	AVL
1000-words.txt	0.04894	0.089558	0.049539	0.088922
5000-words.txt	0.933587	1.710837	0.931266	1.701569
10000-words.txt	0.846074	3.319039	1.847836	3.349868
15000-words.txt	2.475085	3.633799	2.483479	4.528519
20000-words.txt	1.999912	3.710383	2.000826	3.655153
25000-words.txt	2.523766	4.522109	2.466161	4.617755
30000-words.txt	2.897405	5.203438	2.864635	5.211493
35000-words.txt	3.714038	6.729384	3.749462	6.747239
40000-words.txt	4.596915	8.29016	4.623276	8.297459
45000-words.txt	5.59442	10.276259	5.51959	10.229485
50000-words.txt	3.551523	6.484016	3.560035	6.551974

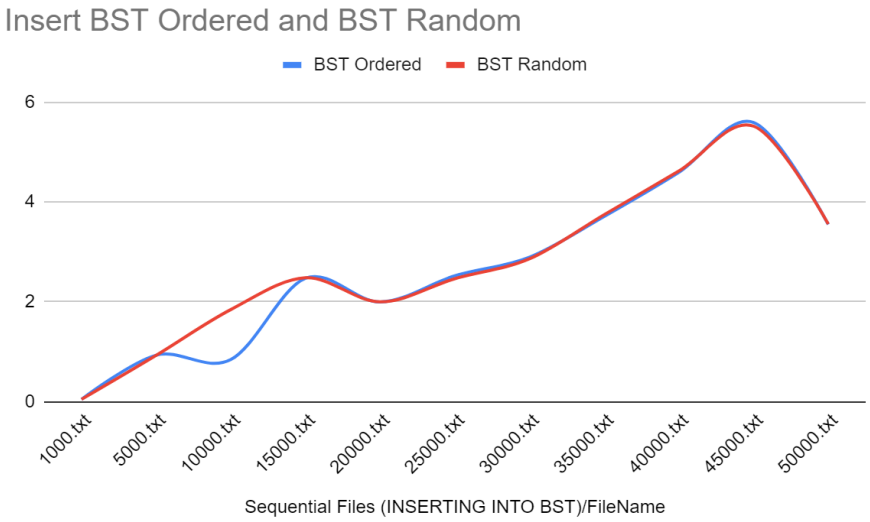
Search Word in Binary Search Tree and AVL Tree				
File Name	Sequential Files		Random Files	
	BST	AVL	BST	AVL
1000-words.txt	0.000482		0.000588	
5000-words.txt	0.000623		0.000424	
10000-words.txt	0.000501		0.000382	
15000-words.txt	0.000534		0.000383	
20000-words.txt	0.00039		0.00039	
25000-words.txt	0.000351		0.000375	
30000-words.txt	0.000361		0.000372	
35000-words.txt	0.000377		0.000381	
40000-words.txt	0.000377		0.000387	
45000-words.txt	0.00037		0.000392	
50000-words.txt	0.000387		0.000398	

Introduction

This Time Complexity Report shows an analysis of two tree data types, BST and AVL. in this study both tree types show their calculatory time of how long it takes to insert a File of increasing sizes, and how long it takes to search for a word. There are 22 files in total that are being used, ranging from 1000 words to 50,000 words, and incrementing up by 5,000 each file. Additionally there are both ordered and random versions of the file to test whether both trees fair better, worse and/or equally depending on the size and randomness of the file used.

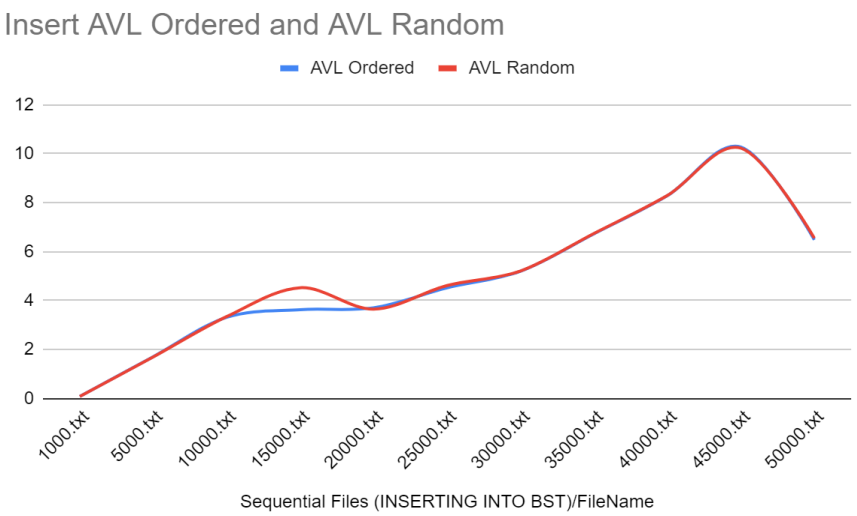
Insert File into BST Random/Ordered

The findings suggest that there is no significant difference of speed between random and ordered files, as the majority of both follow a close to identical trend. This suggests that the order or randomness of a file does not affect the efficiency of a Binary Search Tree process.



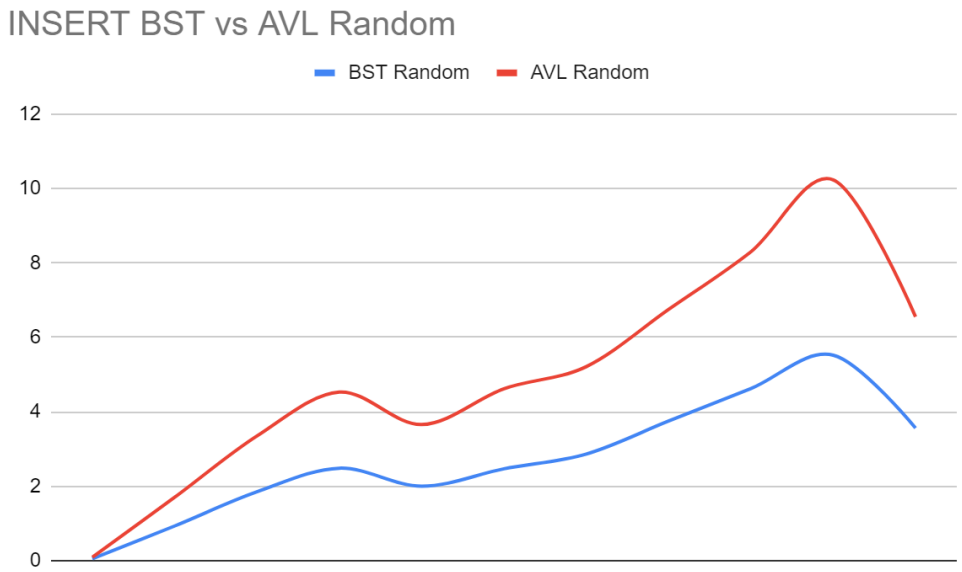
Insert File into AVL Random/Ordered

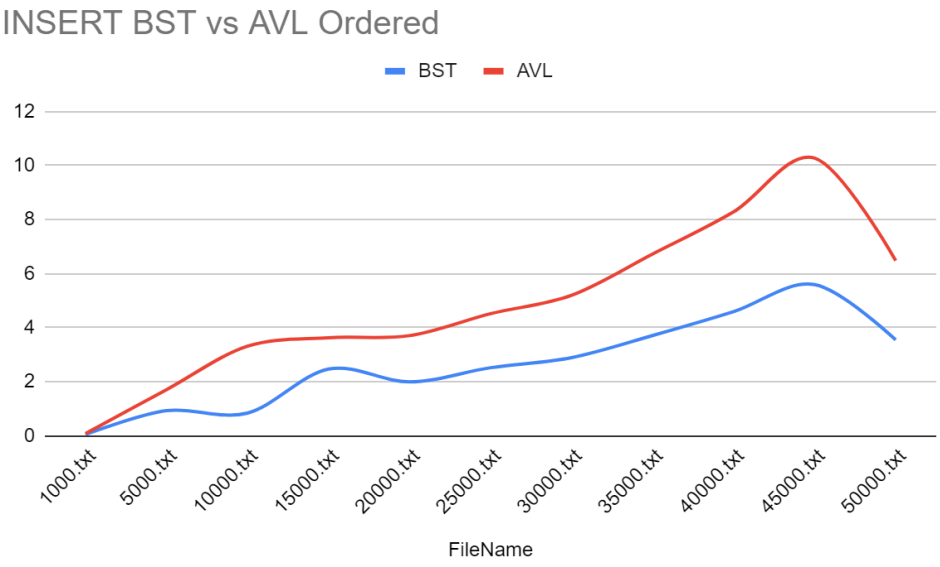
Again much like BST, the ordering of the file does not seem to affect the efficiency of the AVL tree process. In fact AVL seems to be even more closely identical to each other timeframe wise.



Comparison Insert BST vs AVL Random and Ordered

Overall, BST has fared best on both accounts for random files and ordered files. Interestingly though AVL shows a somewhat exponential increase compared to its BST counterpart, with both types having similar starting points with the smaller files, and increasing at larger increments the larger the file is.





Search Word BST vs AVL Ordered and Random

Disclaimer: My Code was not fully functioning on the AVL section, therefore it was not possible for me to complete this section of the Time Complexity Report, however I will still continue my findings for the Binary Search Tree

Through my findings in the BST list, there was a large difference between the two in terms of timeframe during the smaller lists. However as the files grew larger, you can see that the times of both of them gradually lower and flatten out, showing a visual representation of O(Log n) (despite the initial longer wait times in the smaller files). This now tells us that a BS Tree is a very efficient search protocol and should be able to consistently do the same to even larger files.