Martin Bacala

Assignment 5

Binary Search Trees

11 April 2017

Abstract

In Assignment 5, the objective was to scan and read two .txt files, ‘random_dictionary.txt’ and ‘oliver.txt’. For ‘random_dictionary.txt’, we were to sort all the words by the first character of each word and implement them into a series of 26 binary search trees. For example, words starting with the letter ‘c’ would be in the third binary search tree. While for ‘oliver.txt,’ we were to scan the first character of each word and check if that word corresponds with the BST array containing the word. If the particular BST array does contain the word, then the long variables wordsFound and compsFound each go up by one. If the word is misspelled or is not in the dictionary, then the long variables wordsNotFound and compsNotFound each go up by one. For this program, we took the code from Assignment 4 and replaced MyLinkedList with BinarySearchTree. We also used the code that was discussed in the forums on Canvas. In comparison with each other’s outputs, we found that using BinarySearchTree is more efficient than MyLinkedList, as BinarySearchTree found the same total number of words found and not found, 911,672, and 64,139, using a lesser amount of comparisons. Also, the averages compared to one another are different. In Assignment 5, we found an average of 16.0 words found and 11.0 words not found, whereas in Assignment 4, we found an average of 3,544 words and an average of 7,433 words not found. The differences shown in the comparisons is due to Binary Search Tree having a time complexity of O(log n) versus linear search having a time complexity of O(n).

```
run:
Average Number of Words Found: 16.0
Average Number of Words Not Found: 11.0
Number of Words Found: 911672 in 14912145 comparisons
Number of Words Not Found: 64139 in 737186 comparisons
BUILD SUCCESSFUL (total time: 4 seconds)
```