Greg Becker
Programming Assignment 5
11/04/2017

Abstract

The goal of program assignment five was to alter the solution to assignment four. In assignment four each word in an e-book was searched through a dictionary stored in 26 LinkedLists—one for each letter in the alphabet. In assignment five the dictionary is stored in BinarySearchTrees, rather than LinkedLists. For both assignments we had to show the average number of comparisons for words found and not found.

In assignment five we changed the search method for BinarySearchTrees to calculate the number of comparisons made when finding or not finding a word. The alteration consisted of passing an array through the method which would store the number of comparisons. When performing each search, the comparison for words found or not found were added to a variable that would be used at the end to compute the averages. Besides this and changing each instance of LinkedList to BinarySearchTree there were relatively few alterations made to the code.

The results, however, were very different. Far more comparisons were made in assignment four because of the linear design of LinkedLists. To know whether or not a word is contained in a LinkedList one must traverse the complete list, whereas with a BinarySearchTree the number of comparisons required is already split in half when comparing the word in question with the root. This accounts for why assignment four could take a few minutes to produce results while the BST approach required only 11 seconds. To put the boost in efficiency in clear and sobering data: average comparisons for found and not found in assignment five was 16 and 9, respectively, and in assignment four the averages for words found was 3500 and for not found 7401.

```
run:
The average number of comparisons made for words found:
16
The average number of comparisons made for words not found:
9
BUILD SUCCESSFUL (total time: 11 seconds)
```