Description of Data Structures and Optimization

Mikayla Timm

Project 1

In my project, I considered mostly space optimization when implementing my data structures. I open one file at a time and read in the characters to a single character array that I created called masterBuffer. I create a single “Words” struct that holds a 2D character array containing all the tokenized words extracted from the single character array as well as the number of words that were tokenized. I then go through and find the phrases by using strcat to get the number of words I need out of the tokenized words array into a single phrase. I chose to use a Hash to hold the phrases in the file currently being compared against as it provides efficient lookup, and because I already had implemented it last semester. Each phrase is copied into a Node struct that contains the character array “phrase” as well as a left and right node pointer. The right node pointer is the only one that is really used in this project for traversing the buckets in the hash table. Once I hash all of the phrases in the original file, I close it and open each file after that for comparing, one at a time. I use the same masterBuffer each time I read in the characters from the new file, as well as the same Words struct throughout the program. I do free the Hash and all of the nodes each time I start the next set of comparison, however, as this was easier for me to implement.

In general, I would say that my program mostly optimizes for space, as I am reusing the same instances of the data structures for each file comparison, excluding the Hash, and I am closing the files as soon as I finish extracting the characters from each one. This does result in the same files being opened multiple times, however.

If I had to process a significantly larger number of files, I feel my program would slow down significantly, since each time around it has to reopen the files, read all of the characters in, re-tokenize, and re-phrase each time. As I stated before, my data structures optimize mostly for space.