Julia Shea

jts6mq

inlab6

CS 2150

Lab Report

-O2 Flag and Reformatting Output:

Compiling my program with the -O2 flag brought my program's runtime for the 300x300 grid using words2.txt down from 2.3747 seconds to .797988 seconds, and therefore definitively optimized the executable. I ran into some issues using the diff comparator with the provided output vs. my own for 300x300 grid and words2.txt, but after sorting both files, stripping the whitespace, and running the comparator first with smaller files, I was able to get the two outputs to match.

Speed and Big-Theta Runtime:

The average speed of my implementation (run on my own laptop (macOS Sierra) using VirtualBox (Linux Ubuntu)) for the 250x250 grid using words.txt as the dictionary file is 8124 milliseconds. For the 300x300 grid using words2.txt as the dictionary file, the average speed of my implementation is 2895 milliseconds. I'm very surprised that this runs faster than the 250x250 grid, but I would guess it's partly because the table size of my hash table for words2.txt is much larger than the table size for words.txt (size is dependent on the number of words in the dictionary), so this may result in fewer collisions and faster lookup times.

The Big-Theta runtime complexity can be expressed as (rows*columns*words), because the word length is a non-significant constant. The complexity is a result of the nested for loops that iterate through the grid.

Problems Encountered & Shell Scripting:

The biggest challenge I've faced so far in implementing this lab was ensuring that my hash table size was large enough such that words in the dictionary wouldn't map to indices outside the bounds of the table. I tried several methods to resolve this, including (unsuccessfully) trying to rehash with separate chaining and editing my hash function to map to a smaller constraint of inputs (I decided against this as a solution because runtimes were incredibly slow). Ultimately, I decided to make my hash table double the size of the dictionary itself. So far, this has not caused me any problems with either of the dictionaries.

I enjoyed writing the shell script very much. The syntax was easy to learn and is very intuitive, and it seems very useful in computing several runtimes at once. I'd like to learn how for and while loops work to better streamline the code, but for only 5 iterations, writing this specific script was fine. The lab file was very helpful in providing sample code for the script, and it was simple to add the total runtime in milliseconds to the end of wordPuzzle.cpp.