Fr2md Farid Rajabi Nia lab Session 11:00 am

The big-theta running time of your application, and a full explanation as to why.

In order to be consistent in my report, I am only focusing on 300x300 grid and words2.txt as my dictionary. The big-theta running time of my application is #_of_row*#_of_col*#_of_words (rcw).

big-theta = 300 * 300 * 1531

My computer is kind of old and slow and I am hoping that the number that you get will be less than mine. With my implementation of hash function in prelab, my runtime was about 55 seconds. I have done a lot of manipulation with my hash table and its searching factor. I have tried linear and quadratic probing but the run time of separate chaining was still faster than previous two although I assumed that it should not be the case. I might not implement it right, but I got a better result from my separate chaining. I have set my table size to a prime number to increase the efficiency of my table. The table size is the next biggest primer number relative to words counts. It saved around 5 seconds on my average run time. Reduce it to 50 seconds total.

Printing each line of code on screen was another factor that slowed the program. I tried to use a built-in string vector to store the outputted string and then print the vector. This trick saved around 10 seconds. And this point I was standing at 40 seconds.

However, the most significant cut was when I changed got rid of pow in my hash function. I used an array of integers to replace the power function. This trick reduced the runtime around 21 seconds. It was crazy. I was not using a prime number multiplication in my hash function at first and I had so much collisions. But when I introduced the prime number in different powers, it increased the efficiency of my hash function resulting in better timing and less collisions. And I was standing at 19 seconds. However, my average time from averagetimre.sh is 6.3 seconds. So, I am hoping that you use the .sh file to get your final number.

My worst-case scenario was 250x250 grid with words.txt as my dictionary. It took 489 seconds to complete I think I set a record of slowness in my application. Now after the changes, its average runtime is 4.2 seconds.