John Zheng

CS2150

3/7/19

Tuesday, 5pm

Big-theta is the tight bound of a function and it is used most often to compare the runtime of any algorithms. The big-theta running time of my program is bigTheta(r*c*1) where r is the number of rows, c is the number of columns, and w is the number of words. Since the number of words is assumed to be some small constant, we can ignore it. The first two loops depend of the size of the grid, r x c. As the size of the grid increases, the runtime also increases. The last two nested loops do not matter because they only run for a set number of times. They do not scale with the size of the grid.

The time on my original application before any optimization with words2.txt and 300x300.grid.txt is 7.30 seconds. When I tried to increase the time, I changed the original hash function by modding the sum of the ASCII by a fix prime number, in this case 11, instead of the table size. I also removed the power function on line 33. This increased the time to 13.33 seconds on average. This is worse because the sum of the ASCII code is being modded by a small fixed number. That leads to more collision which leads to a large linked list, which is slow to iterate through. To decrease the time by modifying table size, I changed the table size to the next prime number after 10, which is 11. This increase the time to 19.39 seconds on average. The reason being that a small table size means that there is less options for the elements to go, leading to a lot of collisions. That also leads to a large linked list.

To improve my hash function, I removed the power function on line 33. Instead, I just summed up all the ASCII characters and mod it by the table size. This improves the function

because the power function is resource intensive. Also, by increasing the table size, it also decreases the time, but I capped it at the next prime number after 1000. The biggest improvement, however, is adding a prefix hash table to compare the words to. By adding this prefix hash table, the program breaks the loop as soon as the word doesn't match any of the prefix. For example, if the word "building" is in original hash table, then the word Buick would break the loop.