An Huynh

Dhwani Pandya

EECS – 560

February 22, 2019

Lab 4 Report

My program is built based three main classes which are HashTable.h (open hashing), QuadHash.h (Quadratic probing) and DoubHash.h (Double hashing). They are the skeletons for my program and Executive.cpp is where I place all these classes together. In this lab, I used random number generator (rand()) to generate random inputs from 1 to 5m (5,000,015). I also used CPU timing to time the amount of time that each hash table need to insert/find all the inputs to/from the table. Basically, I placed Insert/Find functions between the start clock and end clock for each table, so I can find the amount of time that they take to complete the task. Each table would need to repeat 5 times and store in arrays of size 5 so I can use these data to calculate the average amount of time for these tasks. After 5 repeats, I use other arrays size of 5 to store the average time for each table to access when I need them at the comparison step. Everything will repeat 5 times with different ranges of input (0.1 to 0.5 for insertion and 0.01 to 0.05 for find). Then all the data we need are ready to be used. In the user interaction part, I simply call Print functions to print out the output if the user choose option 1. Finally, I created a table format output on the terminal to represent all the data for comparison. The complexity for open hashing insertion should be O(log(n)) in theory because we simply need to hash to find the location on the table and then add the value to the front of linked list which is O(1). Therefore, the time that need to build and insert values to the table is the slowest among three tables. On the other hand, the complexity for quadratic probing is O(n) because after we got the location on the table, collisions can happen, and we need to do quadratic probing to find a new location until we have one. Double hashing is like quadratic probing in general, because after we found a location on the table and there is a collision, we still need to probe until we have a good location, and the complexity should be O(n). As a result, quadratic probing and double hashing time for building are very similar, but they take longer to insert the value compared to open hashing. The amount of time spends on finding for three table are different. Open hashing has a complexity of O(n\*log(n)) on finding because we first need to find the location on the table, then we travel across the linked list (O(n)) to find the value. Therefore, the amount of time on finding relate to the amount of value in linked list on the location of the open hash table (collisions). On the other hand, quadratic and double hashing finding times are more consistent because the complexity for both are O(n), since we need to travel across the table to find the value. Therefore, open hashing would take longer when we insert a large amount of data compared to the other two tables, and otherwise. These results reflect clearly on my program when I them on multiple size of samples.

In conclusion, we would want to use open hashing when we do not have to insert a large a mount of data because it takes longer when we try to access the data. Unless, your work only required to store data only. On the other hand, quadratic and double hashing are more consistent method to use because it helps us to access and process data faster.