Jean Eyeghe Obame

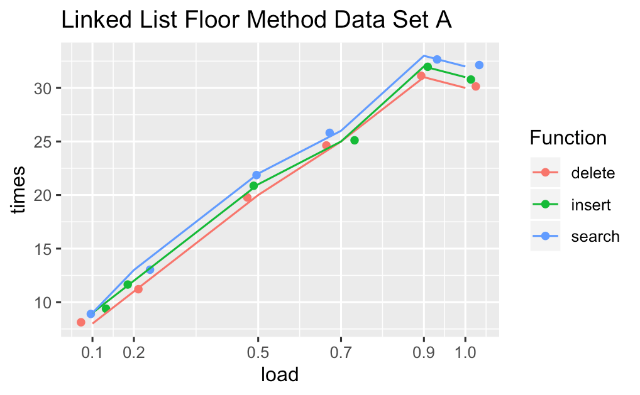
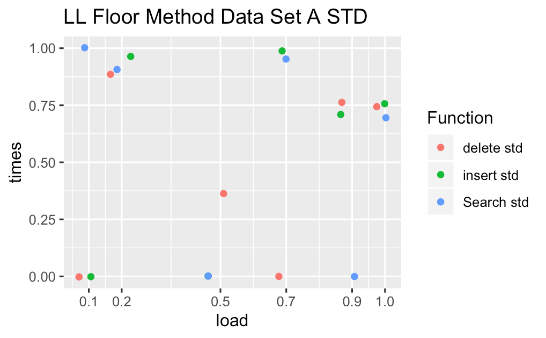
12/8/19

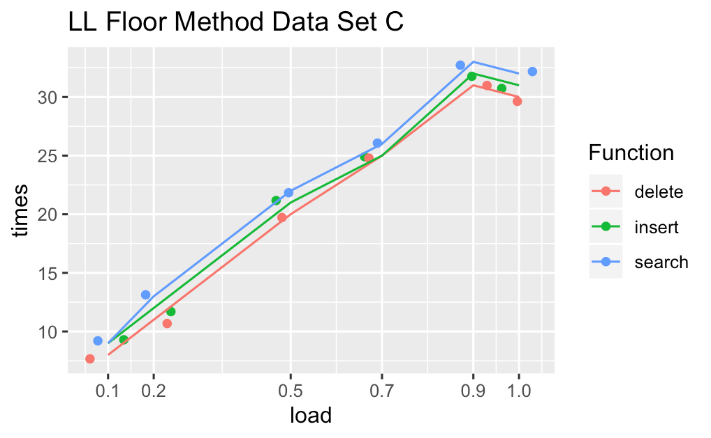
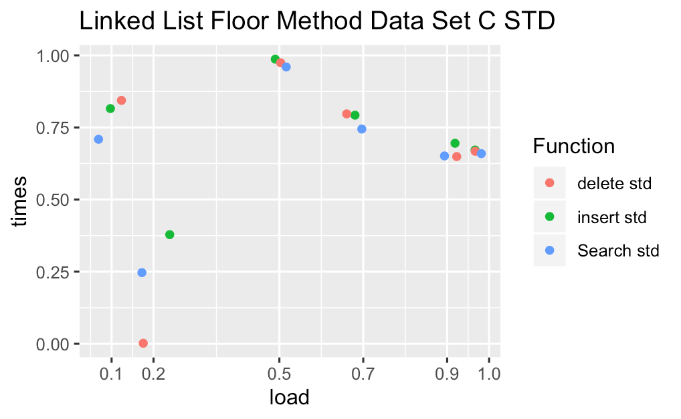
Zagrodski

Final Project Write up: Hashtables

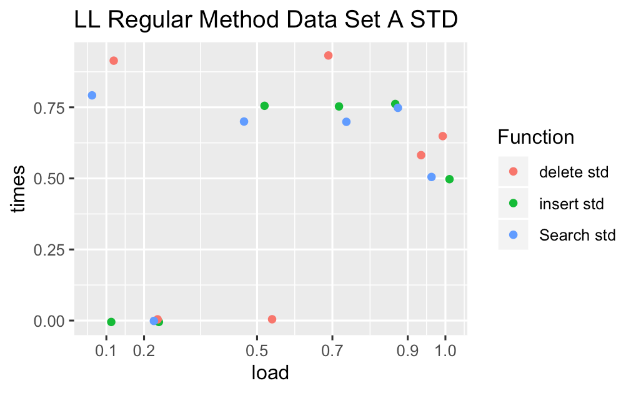
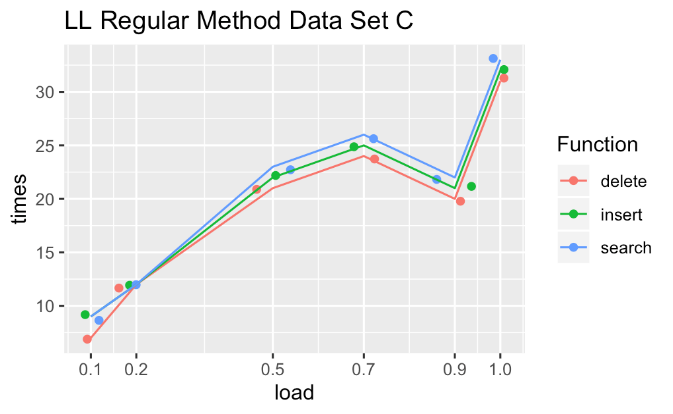
In this project, I collected data from four different variations of hashing: Chaining with a linked list, chaining with a Binary search tree, linear probing, and an all new method learned called Cuckoo Hashing. Both data sets A and C performance are recorded – in average time and standard deviation – by performing delete, insert, and search functions with different load factors of the table to weight out the differences in each load factor step. In this project, I used two methods to hash the number, one regular method (h(x)) and one floor method (h’(x)) and used these to find the differences in performance. Below are the results (in graph form) and our conclusions to the performance of each method, and which is the most efficient. Also attached to this submission will be the tables for all of our results – each hashing method and variation have their own page. (Please note all the units of the data are taken in time of microseconds):

Linked List:



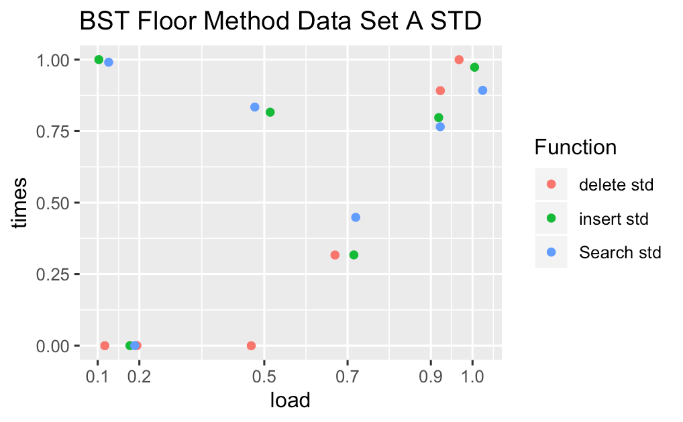
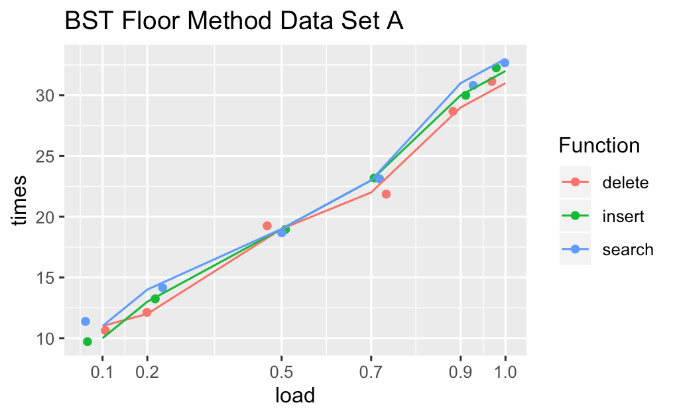


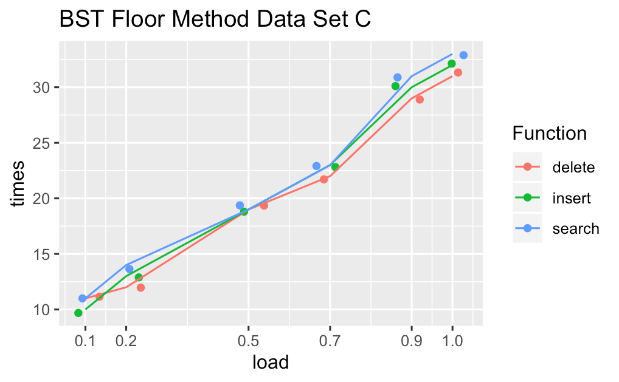
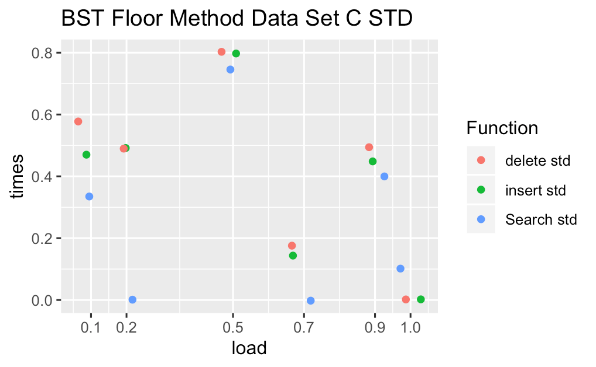
In the floor method (shown above), both data sets yielded a very similar average time. With the lowest load factor, both data sets yielded under seven microsecond function times, and a little over 30 microseconds for full load factor – the standard deviation is quite different for each function instance. The data sets of the floor method are similar to the regular method as well.



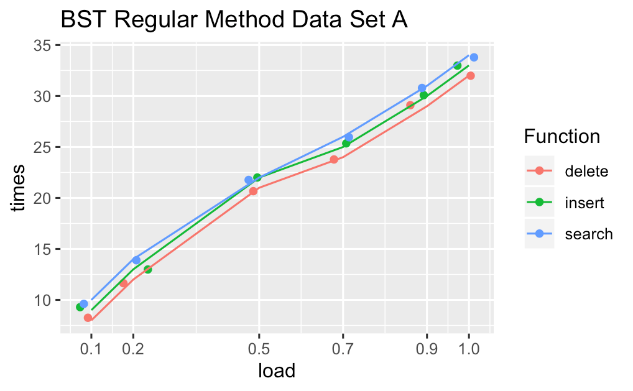
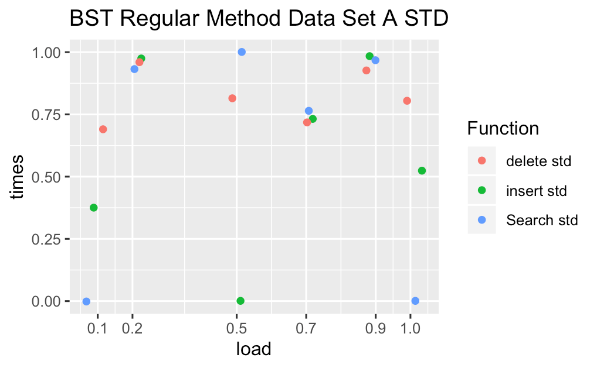


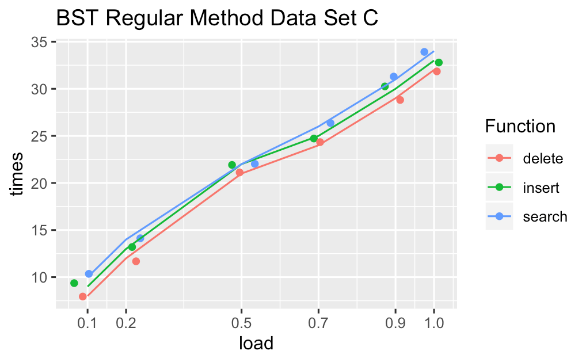
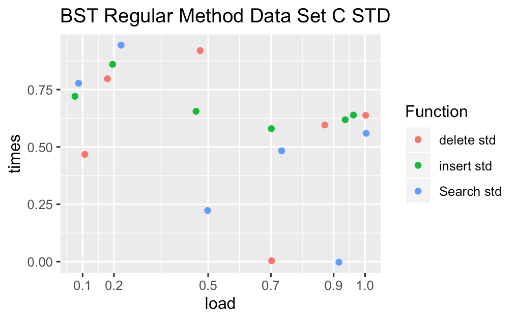
Binary Search Tree:



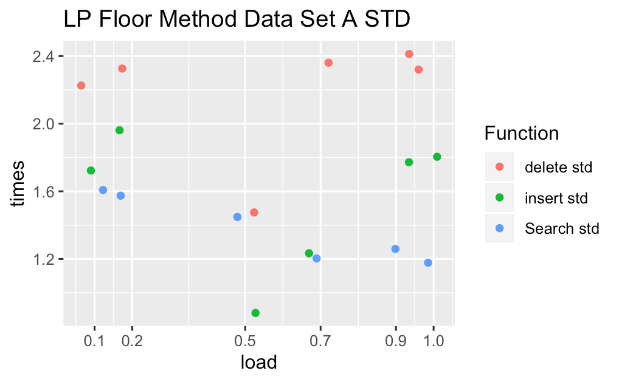


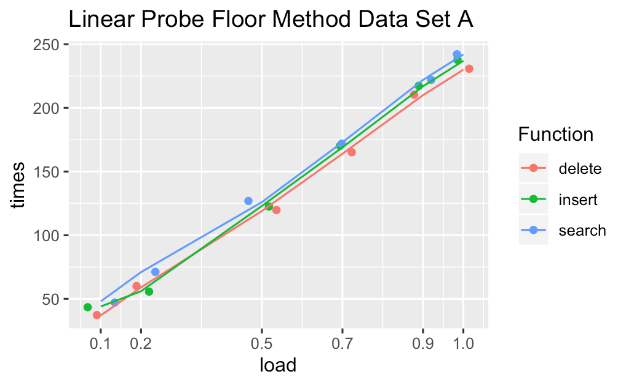
In the Binary Search Tree method (shown above), both sets yielded similar data to the Linked List. The BST had barely higher numbers (about 1 microsecond more each time) for all load factors – the standard deviation is quite different for each function instance as well. The data sets of the floor method are similar to the regular method (shown below) as well.

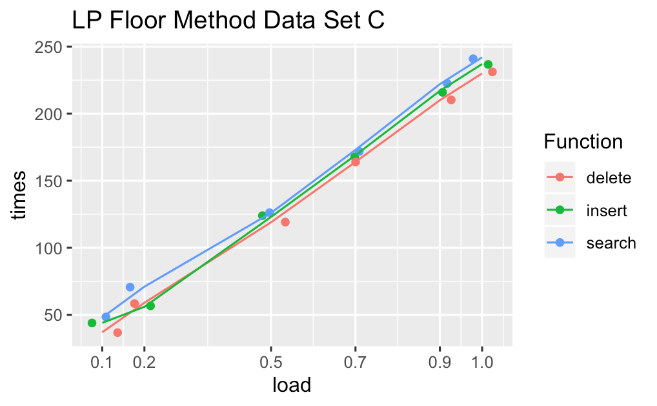
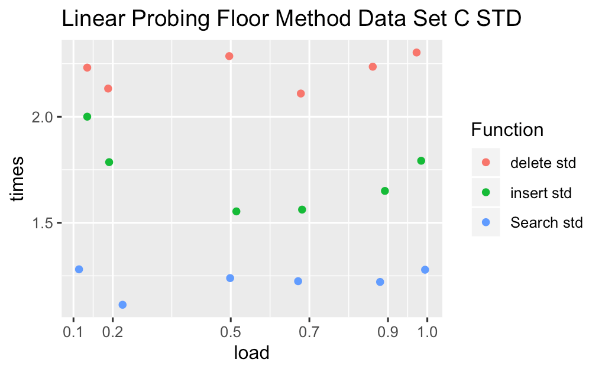




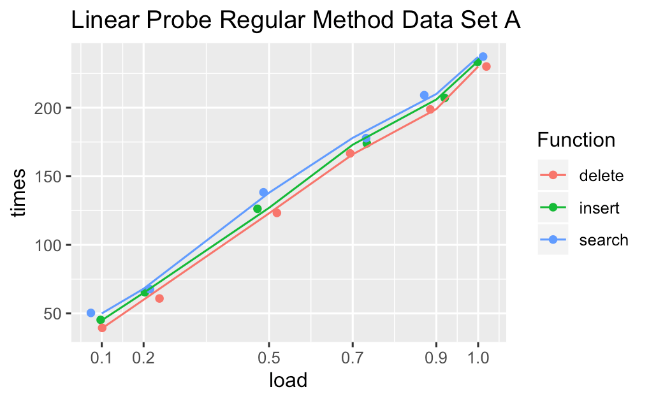
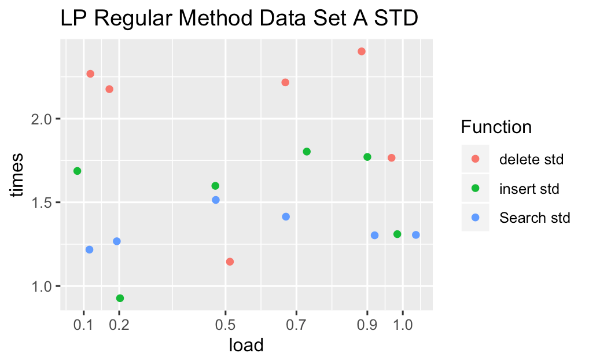
Linear Probing:

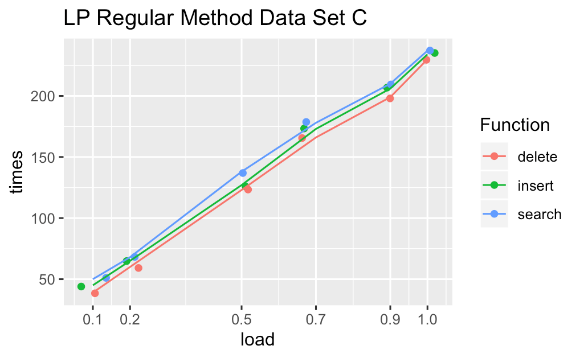
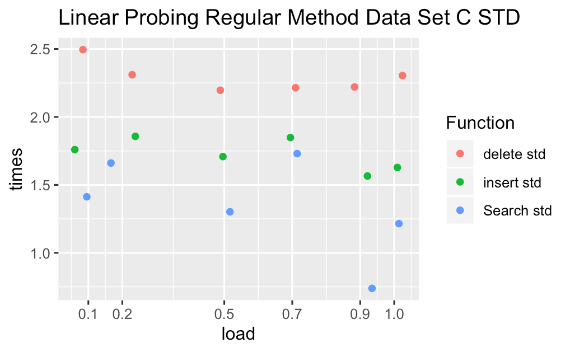




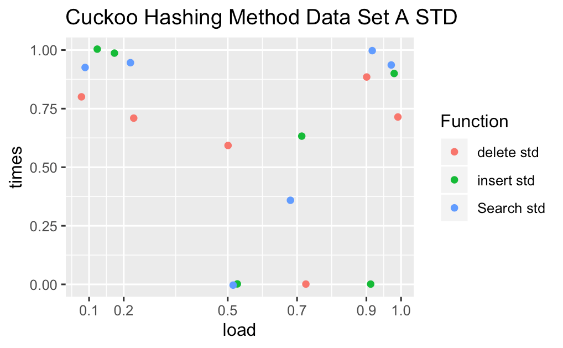


Out of all the variations, Linear Probing yielded the highest average times and standard deviations. Each instance of time was around five or seven times higher than that of Linked List and BST. With a 39 microsecond low and an over 200 microsecond high, this method takes longer – this has to do with linear probing’s method of finding the next open memory slot in the table during hashing. The floor method numbers (shown above) are very similar to the regular method numbers (shown below).



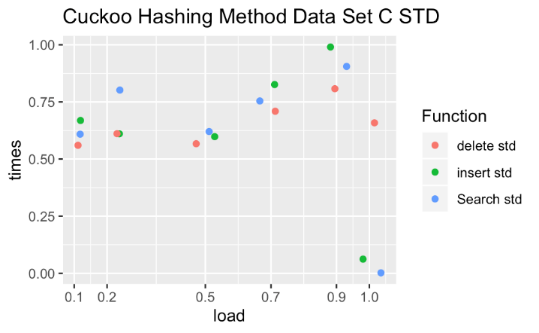


Cuckoo Hashing:



Cuckoo hashing uses both regular and floor methods of hashing, and from the data, this method is fairly efficient. Sitting in between BST and Linked List, it comes very close with very similar times as both of those other methods. The Data for data set A (seen above) is similar to data set C (seen below).





To conclude, each variation was quite impressive in the way it performed, but the efficiency of each can not be said the same. In terms of efficiency, linked list chaining if the most efficient, followed by Cuckoo hashing, Binary Search Tree chaining, and Linear probing. It was very interesting coding and seeing the results of each method and how they come together to do the same objective in different times.