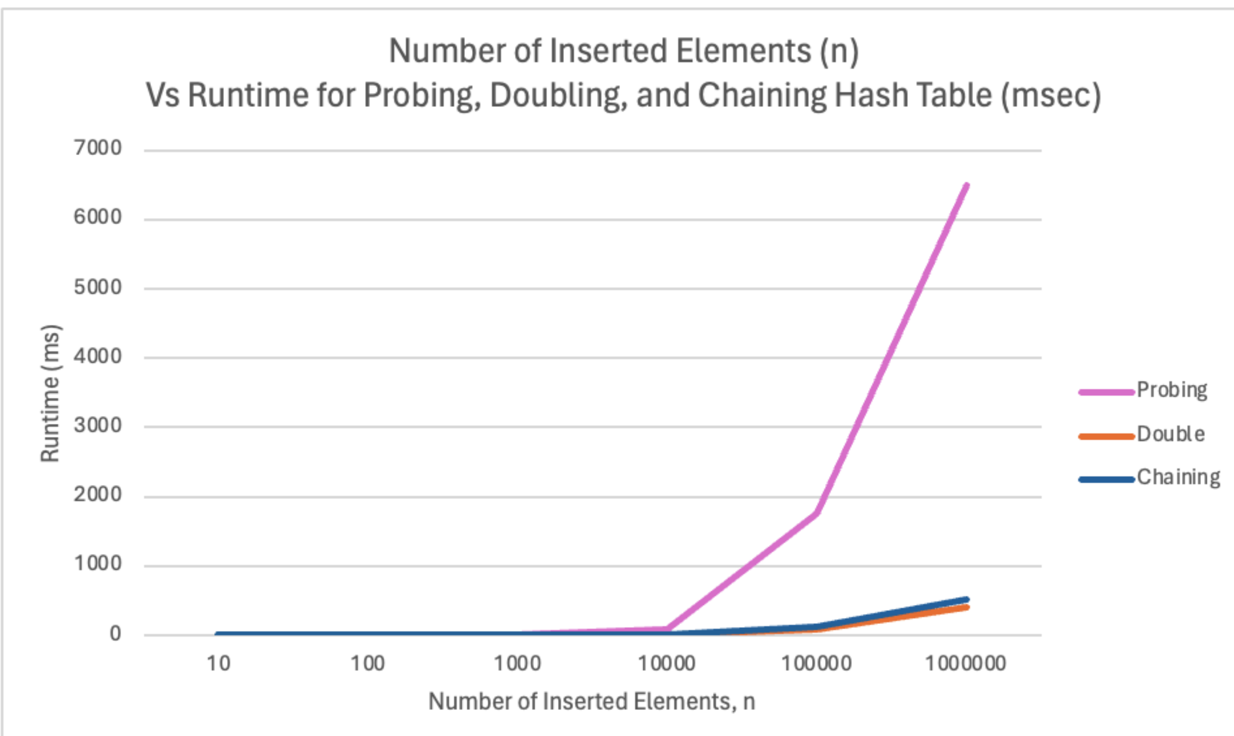
Cody Wu
133001517

CSCE 221: PA4

Theoretical Statement:

Two types of hash tables were implemented. A separate chaining one uses vectors and linked lists to extend the amount of elements at each index. An open addressing one looks for empty spots within a table to insert new elements. A hash table using separate chaining has no amortized runtime and an average time complexity insert of O(1) and a remove of O(n). A hash table using open addressing has an amortized and total time complexity for insert of O(1) and remove of O(n). Runtime depends on the amount of collisions and how many times it is forced to rehash. This usually means that there is an average insert complexity of O(1) but a worst case of O(n).

Experimental Analysis:



The graph above shows that the probing hash table performed the worst while the double and chaining hash tables had very similar results. The double and chaining hash tables also performed the best. The biggest difference began to show at around 10,000 elements.

Discussion:

The separate chaining hash table followed the theoretical O(1). On the other hand, the open addressing hash tables were not the same. The double hash table followed the average theoretical insert complexity of O(1) but the probing hash table did not. This is most likely due to the fact that there were more insertions without deletions. This would eventually cause more and more collisions and rehashing. The reason the probing table had this issue while the double table did not was because of the changing of indices. The probing table was increasing by 1. This meant that elements of the same hash index would have to check the following square. The doubling table rehashed using prime numbers which allowed for more spread and less chances of collision, thus lowering the runtime. The open addressing hash tables are very good about using space. They keep all data close together which optimizes the amount of space they take up. It also allows them to utilize caches very efficiently and reduce time to retrieve data, thus improving their runtimes. Separate chaining does not use the same cache techniques. The data is a lot more spread which can negatively affect their search times.