Ahmdullah Samady- HashTable white paper

For my programming project I chose hash tables as the data structure I would use to implement my program. The reason I chose this programming project case study is because I believe that Hash Tables are a widely used, very popular data structure, and I wanted to gain some experience with them by first understanding their execution details through implementation and then by using them to solve a programming project.

The main advantage of Hash tables is the speed with which it can insert, delete, and count the data that’s stored in it. Because keys are generated through the hashing function, very little irrelevant information is searched for. On average the time complexity for inerst, delete and count in a data structure is O(1), however there are some conditions. A hash table uses generated keys to index each piece of data stored into it. When two “pieces” of data generate the same key then a collision occurs. For my implantation I chose to handle collisions using chain addressing. With chain addressing, a linked list is created at a index where a collision has occurred, and the data that is entered last becomes the last node linked in the link list. If a bad hashing function is used and most of the data is stored in one index, then the time complexity for the Hash Tables methods become O(n) because in general it becomes the same as having a single linked list. When a HashTable has a hashing function that properly organizes data between all the HashTables indexes, the average time complexity is O(1+n/m) where m is the table size and n is the number of items.

Some disadvantages of Hash tables is their memory. Because HashTables initialize the Table with ptrs to items containing data, regardless of how much data will be stored, HashTables have a space complexity of O(n). Compared to other Data structures like binary trees, which only allocate as much memory as they need, Hash Tables can be inefficient when it comes to memory consumption. However, if the amount of data entries is known prior, HashTables remain one of the best data structures for data indexing. Comparing HashTables to other data structures shows that HashTables are in general just arrays that self-organize and which point to linked lists.

Because this project in general called for data indexing, I believe that HashTables were the best data structure for this task. It would have been possible to use other data structures such as trees, but the time complexity of Hash Tables makes it much superior to those data structures in situations such as this where data is being indexed.