Devan O’Boyle

CSE13s

Assignment 7: The Great Firewall of Santa Cruz

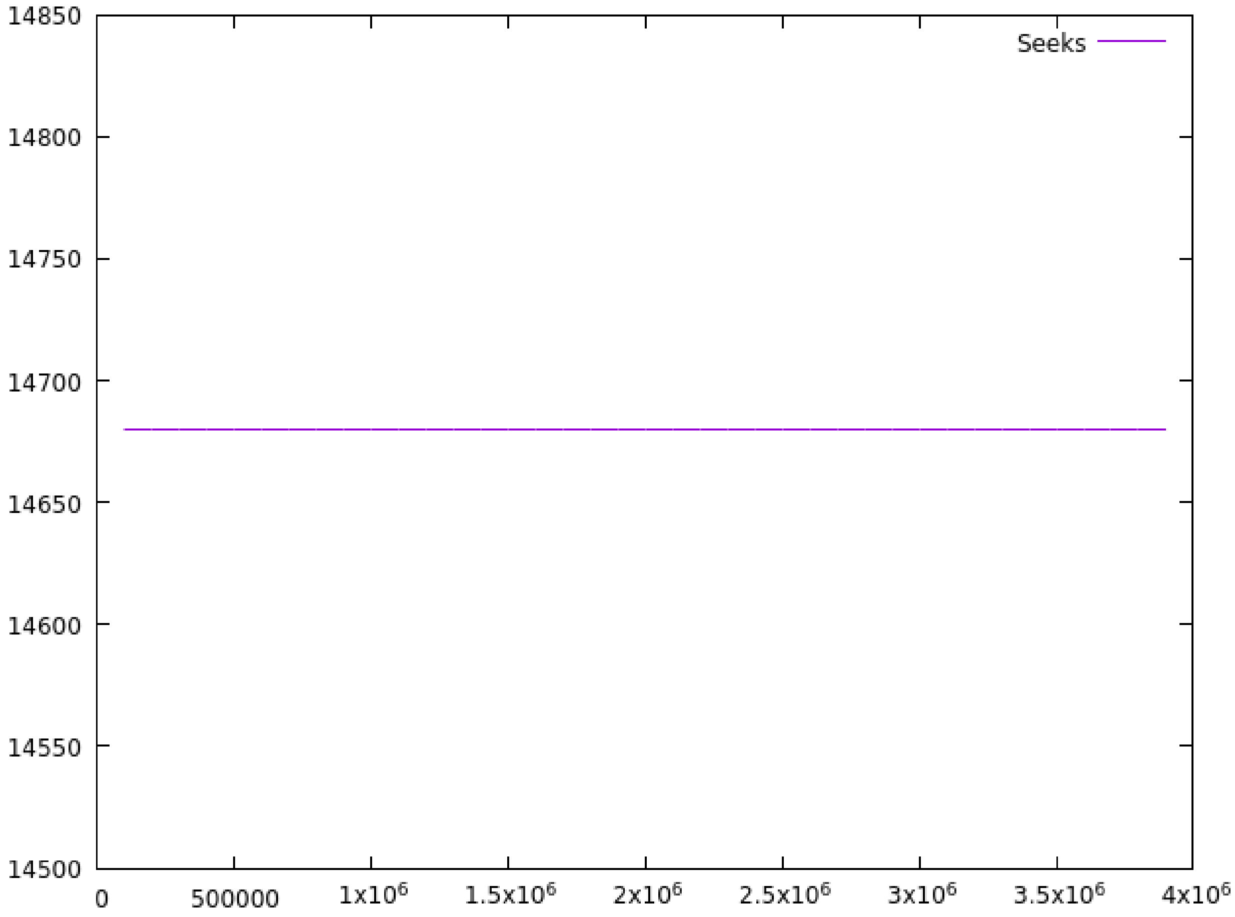
Write Up

In this assignment, hash tables and bloom filters were used in order to check if a word matched another word in the given hash table and bloom filter. The number of times that a lookup was performed on the hash table is the number of seeks. I also measured the average seek length by calculating the total number of links traversed divided by the total number of seeks. Here, I will be looking at the various relationships between the lengths of the hash table and bloom filters to that of the number of seeks, and the average seek length.

*note: all tests performed used the same input file

First, we will look at how increasing the hash table length affects the number of seeks performed.

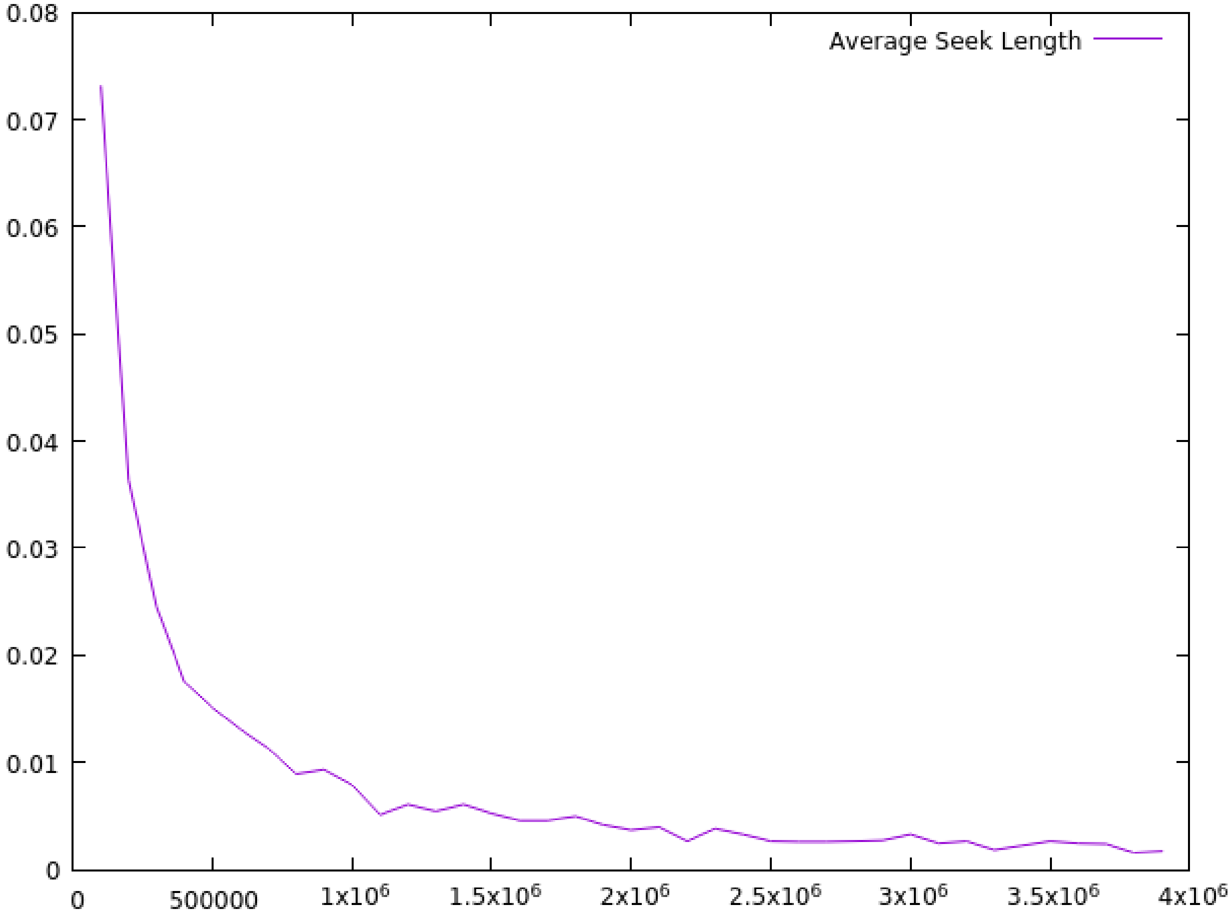
hash table length (x-axis) vs number of seeks (y-axis):



As we can see, the hash table length does not affect the number of seeks because it doesn’t change the amount of lookups required to find a given word.

Next, let’s look at how increasing the hash table length affects the average seek length.

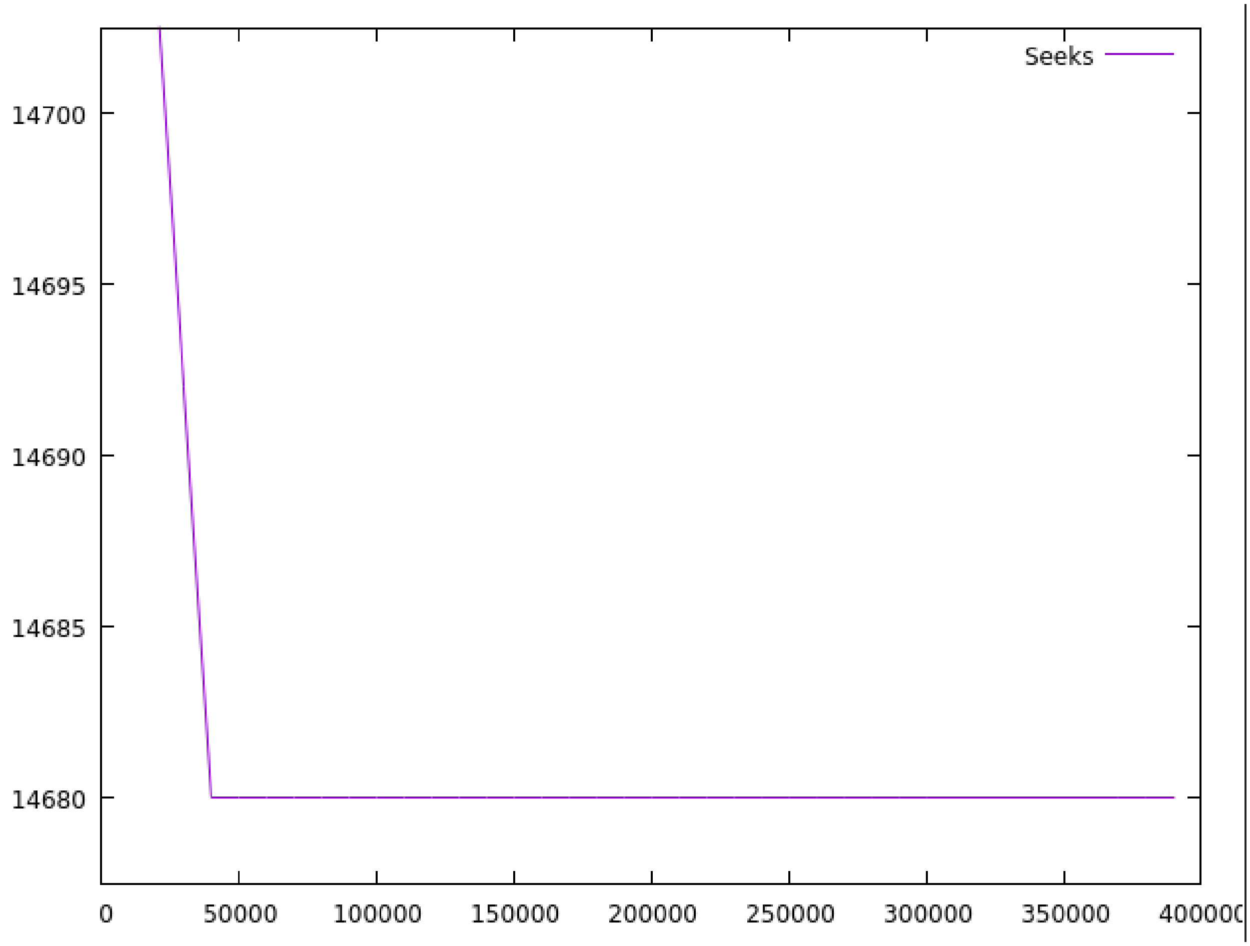
hash table length (x-axis) vs average seek length (y-axis):



We know that seeks stay the same, so therefore, the number of links is decreasing as the hash table length increases. This is because the hash table can have more linked lists so the hash table is able to spread the items among more lists which causes there to be less items in each list. Therefore, there are less links in each list meaning that the linked lists get shorter as the hash table size increases.

Now take a look at how the number of seeks changes as we increase the bloom filter length.

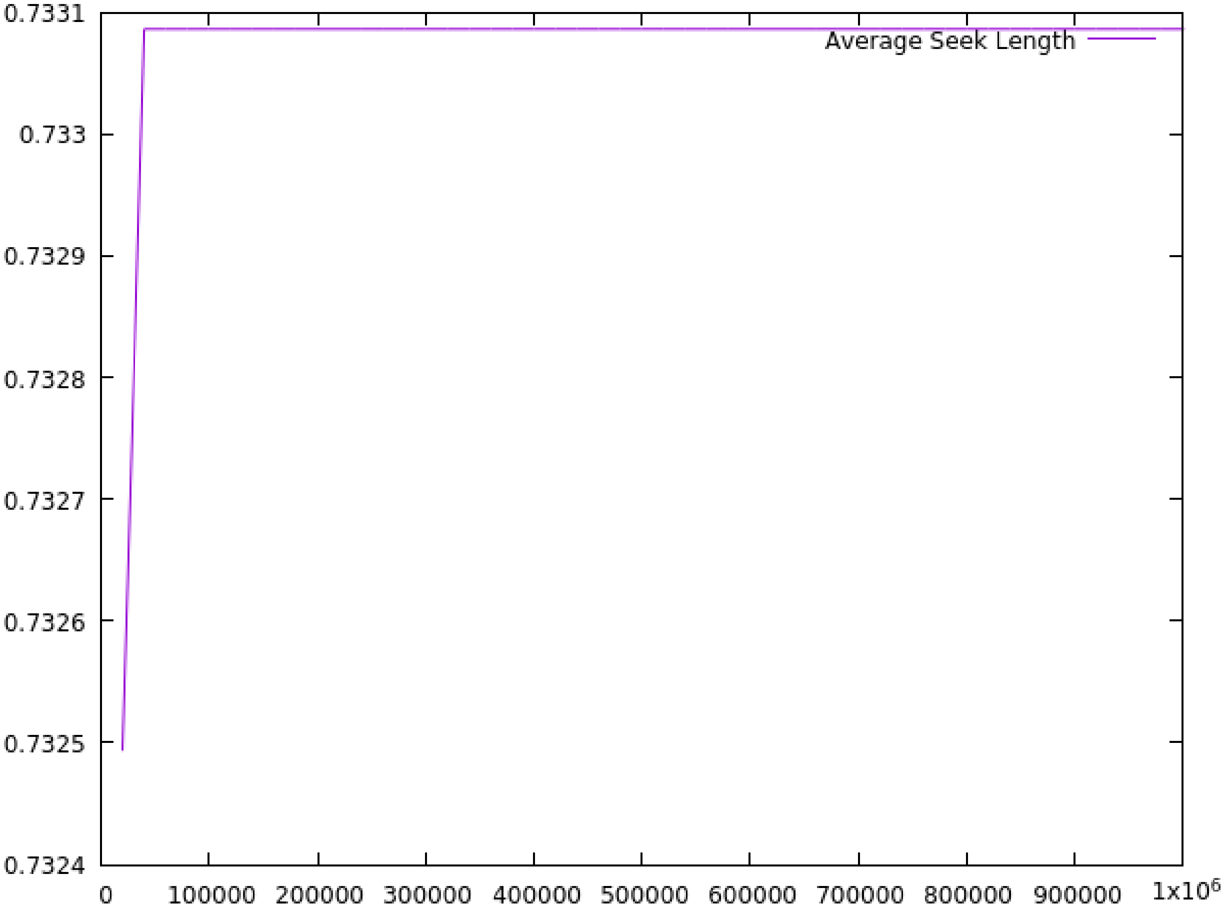
bloom filter length (x-axis) vs number of seeks (y-axis):



As we can see, the number of seeks is larger at lower lengths of the bloom filter and then quickly has the same number of seeks at higher bloom filter sizes. This is because the likelihood of false positives is much higher when the bloom filter is smaller. So once the bloom filter size got large enough and was able to run without false positives, we then end up with the same amount of seeks for all the rest of the larger sizes since the likelihood of getting a false positive is so low.

Take a look at how the average seek length changes as the bloom filter length is increased.

bloom filter length (x-axis) vs average seek length (y-axis):

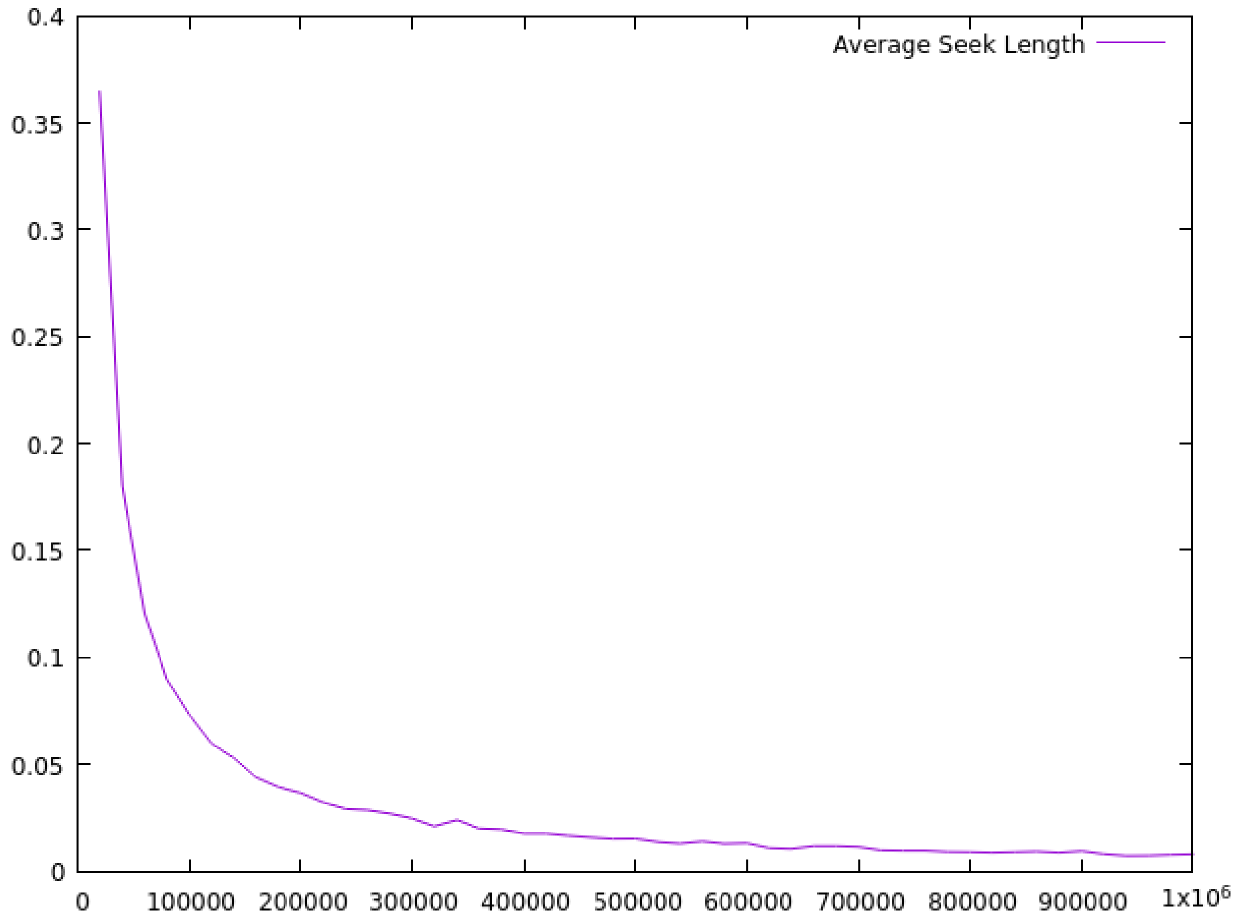


Here, a very similar principle applies to the number of seeks graph from before. Since the bloom filter is more likely to have false positives at lower size values, therefore there won't be as many seeks at those lower values. However, once we reach a high enough value, there eventually won't be anymore false positives allowing the graph to flatten out as it did with the previous graph.

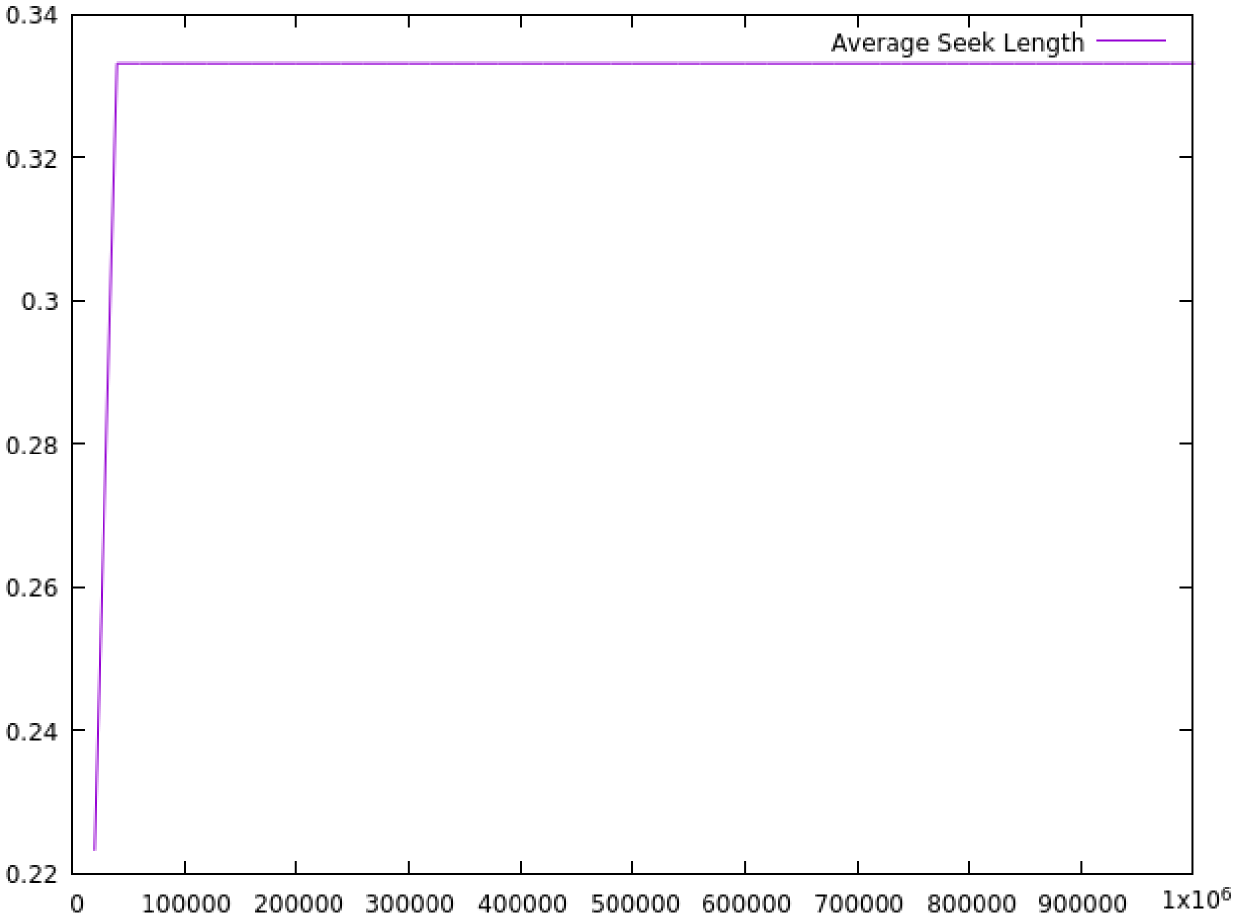
Now let's take a look at the graphs displaying the number of links for both the hash table and the bloom filter, but this time with the move to front option enabled.

with move to front:

size of hash table (x-axis) vs number of links (y-axis):



size of bloom filter (x-axis) vs average seek length (y-axis):



As we can see for both of these graphs the relationships of the graphs stayed the same but the average seek length decreased drastically. This is because the move to front rule makes it so less links have to be traversed in order to find the more common words. The more common a word is with move to front protocol, the more likely it will be near the front of the linked list. So less links will be traversed on average for each seek. Since the same amount of seeks are performed with or without move to front enabled, the average seek length decreases substantially.