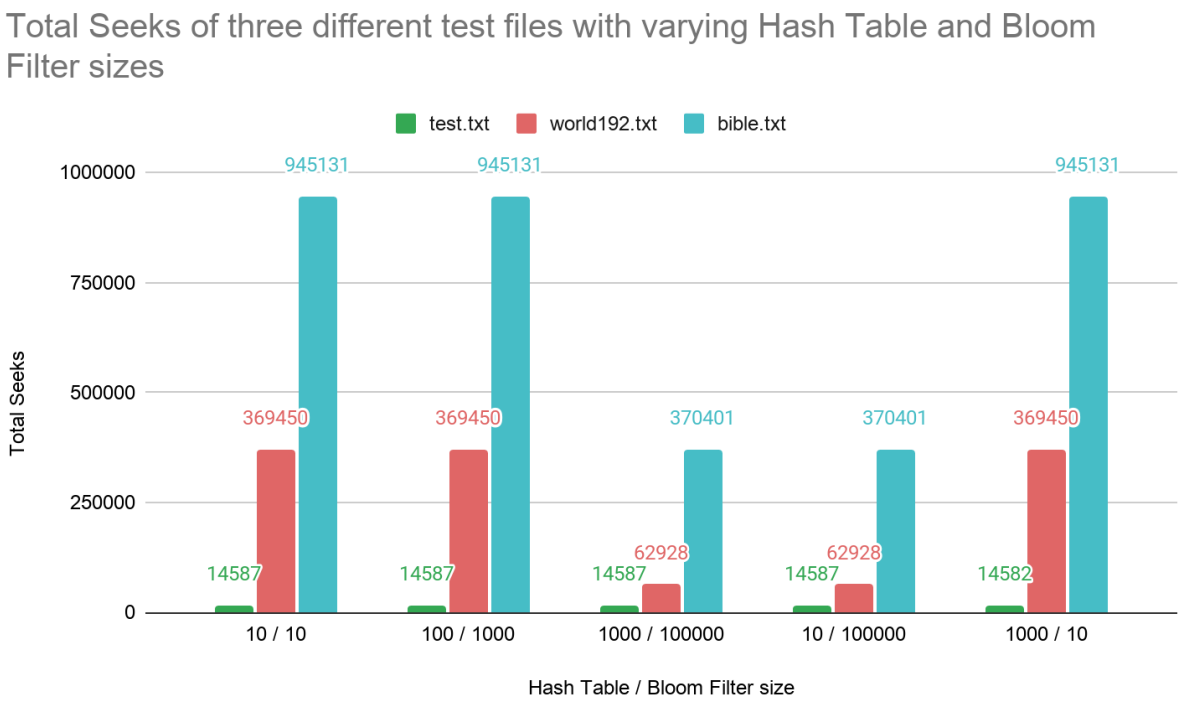
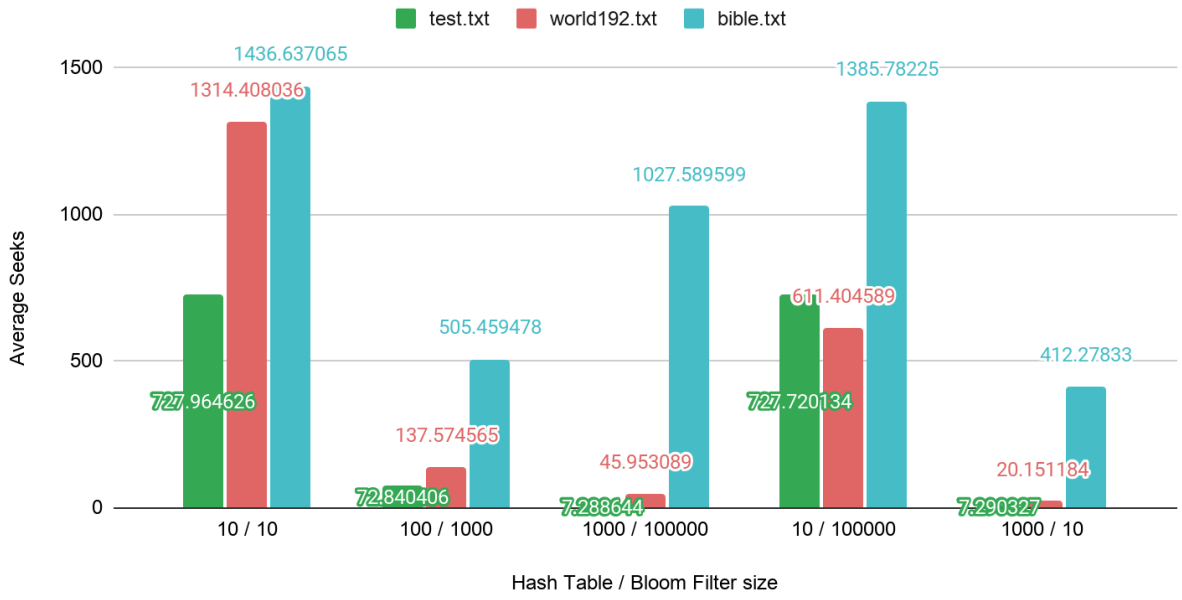
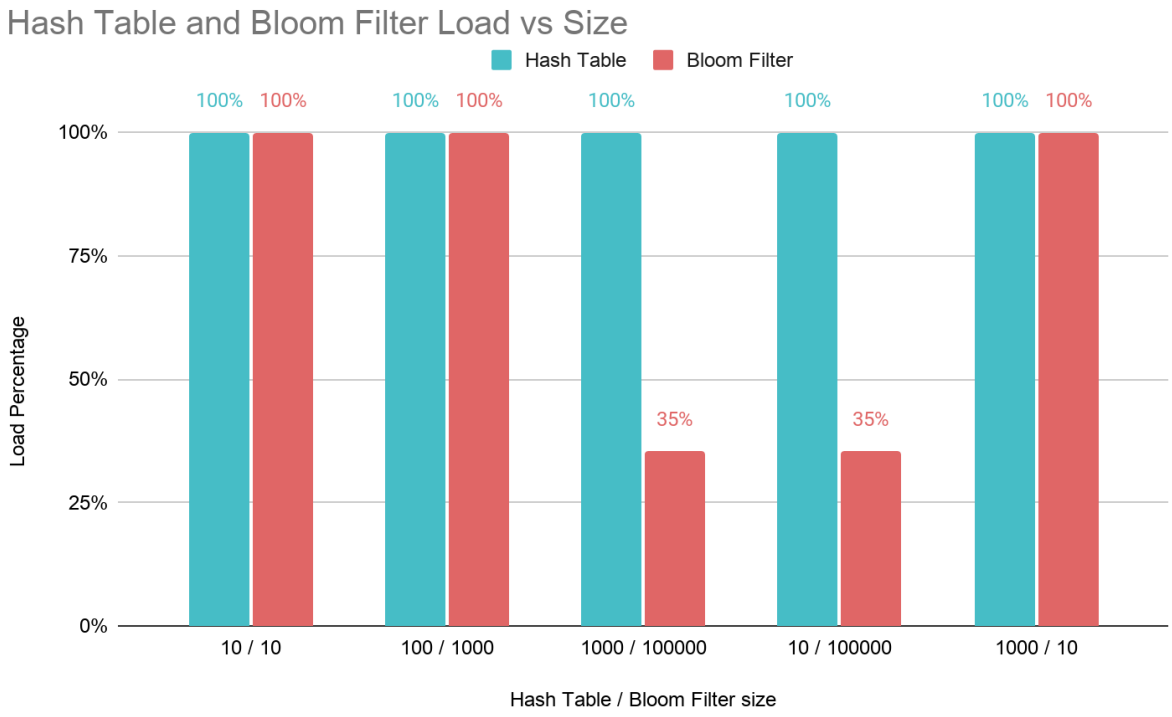
Ravjodh Heer

Assignment 7 - The Great Firewall of Santa Cruz

Graphs:







Analysis:

These three graphs show data gathered between three files, one small custom file created by myself with a 9 word sentence containing 4 badspeak/newspeak words, and two large files containing many badspeak/newspeak words. I recorded the total number of seeks, average seek length and the Hash Table and Bloom Filter load percentages for each different set of Hash Table and Bloom Filter sizes. I chose to record data and compare them between five different Hash Table and Bloom Filter sizes in order to detect how a change in either size can affect any element of my data. The first graph that records the total number of seeks stayed the exact same throughout all sizes except when I increased the Bloom Filter size to 100,000. I believe that the reason why increasing the bloom filter size would cause the number of seeks to decrease would be because there's also a decrease in Bloom Filter load as shown on the third graph. Why then, are the other 3 values for each file completely identical even though I varied the

Bloom Filter size from 10 to 1,000 to 100,000? This could be possibly due to the fact that the Bloom Filter load determines the number of seeks that the program performs and only when the load decreases, the number of seeks would decrease. Why then, is the Bloom Filter load not affected by a size change from 100 to 1,000 but is affected when the size changes from 1,000 to 100,000? I believe this could be due to the fact that the Bloom Filter sizing operates exponentially since the default value given to us was 2^20 and a change from ~2^7 to 2^10 (~100 to ~1,000) would not affect the load on the Bloom Filter as much as 2^10 to 2^17 (~1,000 to 100,000) would. In our case, a 10x increase in Bloom Filter size amounted to a 3x decrease in Bloom Filter load. This in turn led to a 5x decrease in total seeks for the smaller of the "large" files and a 2.5x decrease in total seeks for bible.txt, the largest file. This effect on total number of seeks however, only seemed to affect larger files since the smaller, one sentence file seemed to maintain the same number of total seeks throughout each test. The data for the average seek length however, was a completely different story from the data for the total number of seeks. This can be attributed to the fact that the average seek length relies on both the total number of seeks and the number of links traversed. Therefore, seeing as the average seek length varied slightly through the five different inputted Hash Table and Bloom Filter sizes, it can be inferred that the number of links is affected by the variance in sizes. From my observations, I hypothesize that the number of links increases equally to the increase in Hash Table size. This hypothesis comes from the fact that the equation for calculating the average seek length is: seeks/links; meaning an increase in links would result in a decrease in the resulting average seek length. As shown by the first three Hash Table sizes of 10, 100, and 1000, you can see that as the Hash Table size is increased by a multiple of 10, the average seek time is decreased almost equally by 10.

As the Hash Table size increased from 10 to 100 to 1000, the average seek length of the test.txt file went from ~728 to ~72.8 to ~7.28. Almost an exact multiple of 10 between the three values. For larger files however, this was not exactly the case. For larger files, seeing as the Bloom Filter load changed the number of seeks when the Bloom Filter size was increased to 100,000, it could not be reasonably concluded that a similar pattern exists. In conclusion the sizes of both the Hash Table and Bloom Filter play a large role in determining the loads of each which in turn determine the number of seeks and links. If anything the above provided data on the graphs can comfortably conclude and support that thesis.