CSE 13S Assignment 6 - The Great Firewall of Santa Cruz

Jaren Kawai - jkawai

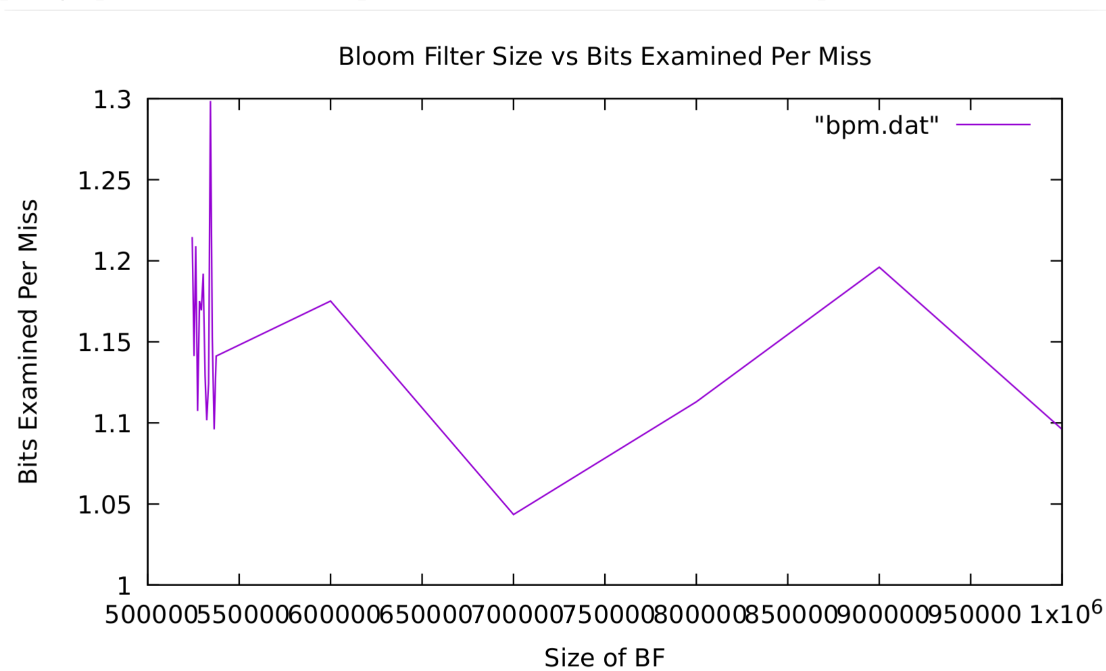
What was Learned

In this assignment I learned more about hash tables, linked lists, bloom filters, and bit vectors. Although we are formally taught about these topics in lecture, I am a more hands-on learner, and the things taught in class began to make more sense when I was actually attempting to implement these data structures.

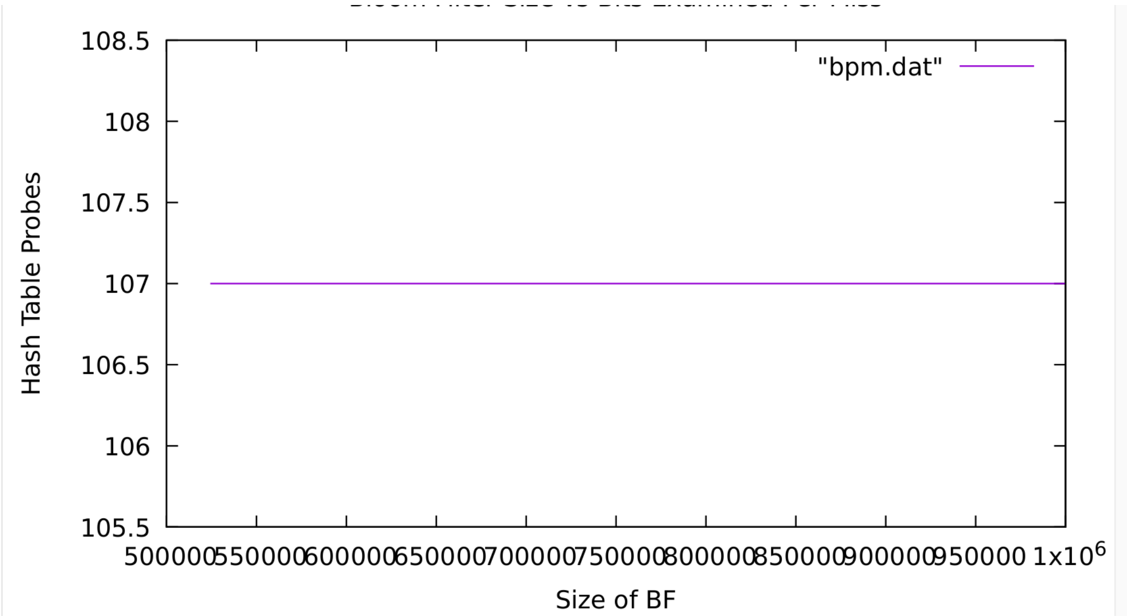
In addition, topics that I didn't necessarily understand at first began to make more sense as I went through the project because when I would code something based on what I knew and it didn't work, I had to go through the lecture slides and try to figure out my problems again until I found something that worked. Although I spent considerable amounts of time on this, I came out with a far greater understanding of these topics that can be taken to other classes.

Further Analysis

From the graph, it seems that the number of bits examined per miss was around the same amount for the majority of the tests. The earlier part of the graph represents minor changes to bloom filter size while that later represents more vast changes, and it seems that although the bits examined per miss could be lower, all values are within a bit of each other meaning that there is no significant change. Each test was run by inputting a text file which contained the first two paragraphs of 1984, and outputted the amount of bits examined per miss.



When changing the size of the bloom filter and looking at the number of hash table probes, I found that for all the values of hash table size that I ran, the amount of probes remained the same. I even tested this against the dist file given, and found that the values that it produced were the same as mine. Changing the bloom filter size can reduce the amount of false positives as the likelihood of finding a set bit decreases because there are more possible places for a bit to be. In addition, having a good hash function can allow for more “random” bits to be set which could also reduce the likelihood of having a false positive in a bloom filter.



From looking at statistics run with different input files, using move to front would reduce the amount of probes in the hash table, which is related to the amount of links traversed in a linked list. This is because when we use lookup, we are moving the found node to the front, so in terms of iterations, fewer nodes have to be iterated over to find desired nodes. Hence, when there are a lot of elements or nodes in a linked list, moving to the front makes finding nodes easier, and by association can make the process of inserting a node into the linked list faster.

The reason why graphs were used here is to eliminate the chance of having confounding variables which are variables that can hinder the ability to see or prove association.

Here, three different input files are used: one with a Shakespearean play script, one with the entirety of 1984, and another with random text. Each test involved running banhammer with both mtf enabled and disabled, and as the pictures show, mtf resulted in fewer probes.

```
Bloom filter load: 0.129885
jkawai@jkawai-VirtualBox:~/cse13s/cse13s/asgn6$ cat "1984.txt" | ./banhammer  -s
ht keys: 14678
ht hits: 60
ht misses: 0
ht probes: 107
bf keys: 14678
bf hits: 60
bf misses: 177
bf bits examined: 515
Bits examined per miss: 1.214689
False positives: 0.000000
Average seek length: 1.783333
Bloom filter load: 0.129885
jkawai@jkawai-VirtualBox:~/cse13s/cse13s/asgn6$ cat "1984.txt" | ./banhammer -m -s
ht keys: 14678
ht hits: 60
ht misses: 0
ht probes: 70
bf keys: 14678
bf hits: 60
bf misses: 177
bf bits examined: 515
Bits examined per miss: 1.214689
False positives: 0.000000
Average seek length: 1.166667
Bloom filter load: 0.129885
jkawai@jkawai-VirtualBox:~/cse13s/cse13s/asgn6$
```

```
jkawai@jkawai-VirtualBox:~/cse13s/cse13s/asgn6$ cat "test.txt" | ./banhammer -s
ht keys: 14678
ht hits: 60
ht misses: 0
ht probes: 107
bf keys: 14678
bf hits: 60
bf misses: 177
bf bits examined: 515
Bits examined per miss: 1.214689
False positives: 0.000000
Average seek length: 1.783333
Bloom filter load: 0.129885
jkawai@jkawai-VirtualBox:~/cse13s/cse13s/asgn6$ cat "test.txt" | ./banhammer -m -s
ht keys: 14678
ht hits: 60
ht misses: 0
ht probes: 70
bf keys: 14678
bf hits: 60
bf misses: 177
bf bits examined: 515
Bits examined per miss: 1.214689
False positives: 0.000000
Average seek length: 1.166667
Bloom filter load: 0.129885
jkawai@jkawai-VirtualBox:~/cse13s/cse13s/asgn6$
```

```
Bloom filter load: 0.129885
jkawai@jkawai-VirtualBox:~/cse13s/cse13s/asgn6$ cat "s.txt" | ./banhammer -s
ht keys: 14678
ht hits: 5046
ht misses: 0
ht probes: 7999
bf keys: 14678
bf hits: 5046
bf misses: 11493
bf bits examined: 38682
Bits examined per miss: 1.170452
False positives: 0.000000
Average seek length: 1.585216
Bloom filter load: 0.129885
jkawai@jkawai-VirtualBox:~/cse13s/cse13s/asgn6$ cat "s.txt" | ./banhammer -m -s
ht keys: 14678
ht hits: 5046
ht misses: 0
ht probes: 5251
bf keys: 14678
bf hits: 5046
bf misses: 11493
bf bits examined: 38682
Bits examined per miss: 1.170452
False positives: 0.000000
Average seek length: 1.040626
Bloom filter load: 0.129885
jkawai@jkawai-VirtualBox:~/cse13s/cse13s/asgn6$
```

From the graph, it shows that in general, having a larger hash table would reduce the amount of probes. However, at a certain amount of time, the hash table's size seemed to remain the same. Like before, more minor changes were made to the hash table's size, and the amount of probes eventually decreased. However, with more drastic changes, the amount of probes seemed to remain the same.