Jack Campbell

Assignment 7: Writeup

VARYING HASH TABLE SIZE

When testing the total number seeks and average seeks lengths as hash table and bloom filter size varied, I decided to keep one of the two constant as I changed the other. When keeping a constant hash table size, I found that both the number of seeks and their average length increased exponentially with a decrease of bloom filter size. The data I used to create the graphs is below.

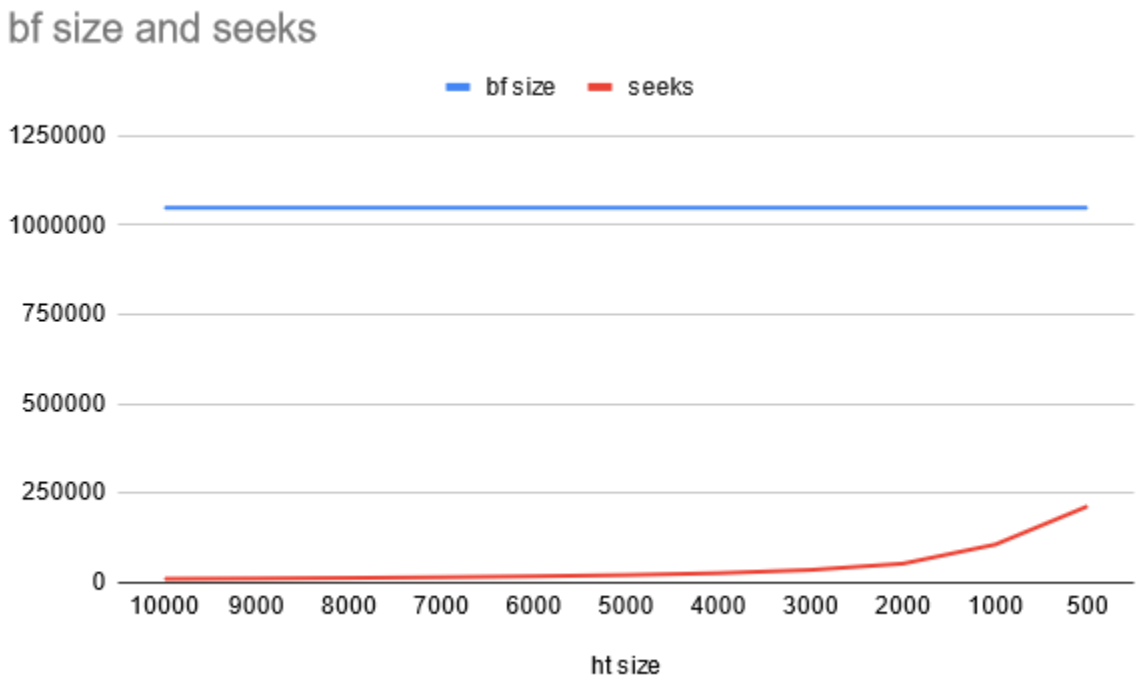
HT SIZE	BF SIZE	SEEKS	AVERAGE SEEK LENGTH
10000	1048576	10781	0.733
9000	1048576	11849	0.805
8000	1048576	13318	0.905
7000	1048576	15231	1.035
6000	1048576	17941	1.22
5000	1048576	21409	1.456
4000	1048576	26604	1.805
3000	1048576	35658	2.425
2000	1048576	53267	3.623
1000	1048576	106764	7.266
500	1048576	213806	14.543

custom text file

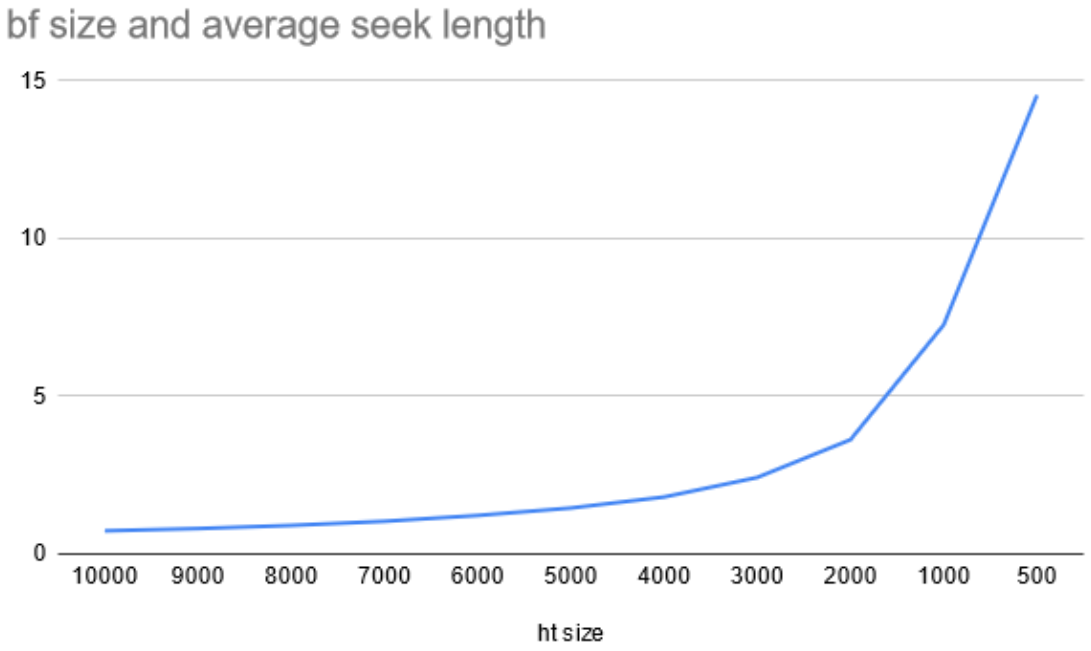
The input file used was a small text file of 10 to 12 words. It was only after I had created the graphs that I realized it may have been better to use a larger text file, but, given a correlation was found, the data was used. That is not to say that the input file was optimal to accurately represent the correlation between bloom filter size and the number of seeks / average seek length. Because it was such a small file, the hash table may not have been affected by the decrease of bloom filter size— as it would have been given a larger file. There is also a possibility that file size had no effect on the resulting number of seeks / average seek length because the hash table load had reached 100% regardless.

VARYING HASH TABLE SIZE: GRAPHS

y-axis: bf size



y-axis: average seek length



VARYING BLOOM FILTER SIZE

When testing the total number seeks and average seeks lengths as bloom filter size varied, I decided to use a larger file: bible.txt. When keeping a constant hash table size, I found that both the number of seeks increased exponentially with a decrease of bloom filter size. Interestingly, the average seek length seemed to stay relatively constant as size decreased until it began to decrease near the end. This makes me think that average seek length could potentially be a negative exponential. Building off of this potential explanation, the decrease in average seek length and total seeks performed seems to be inversely proportional; I decided to test it.

SEEKS	AVERAGE SEEK LENGTH	SEEKS * AVERAGE SEEK LENGTH
343468	1104	379188672
343468	1104	379188672
343468	1104	379188672
343498	1104	379188672
344114	1102	379213628
366106	1036	379285816
413082	918	379209276
616343	616	379667288
788396	482	380006872
799603	475	379811425

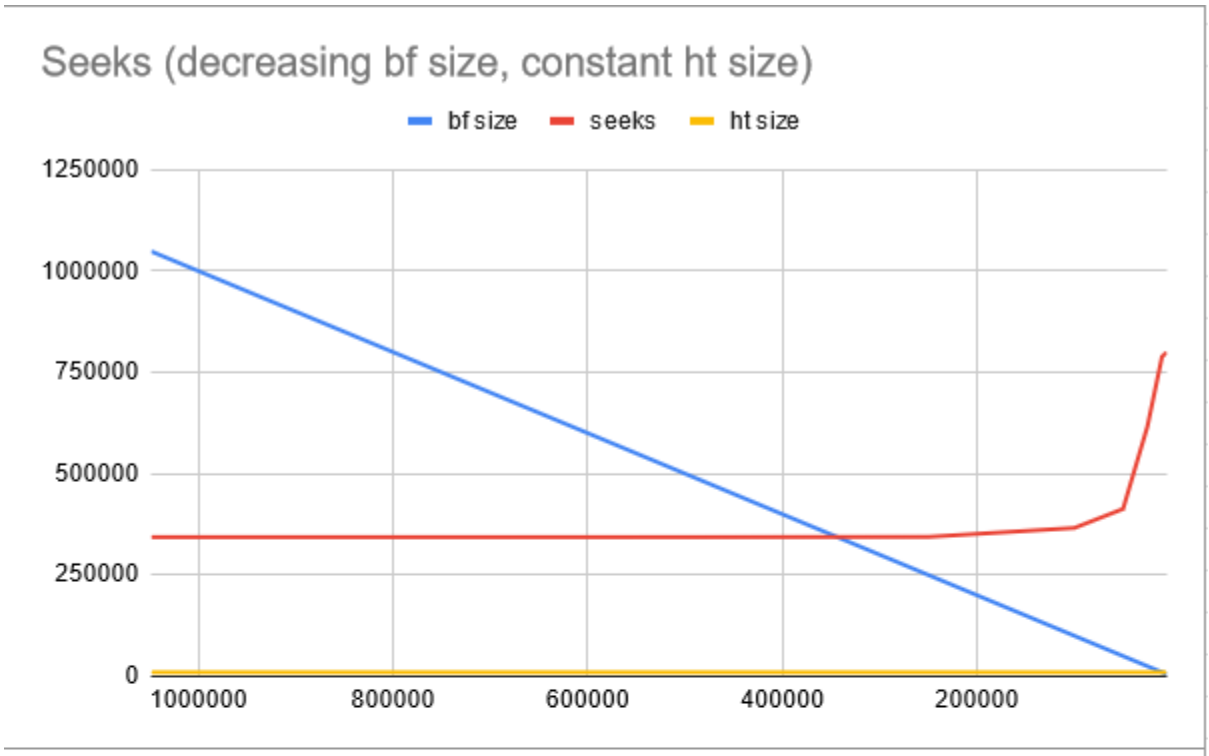
In the chart above, you can see that the product of the total seeks and the average seek length is relatively consistent despite both of the elements changing. This makes me believe that the same amount of nodes were iterated over, but it was done in more, smaller chunks.

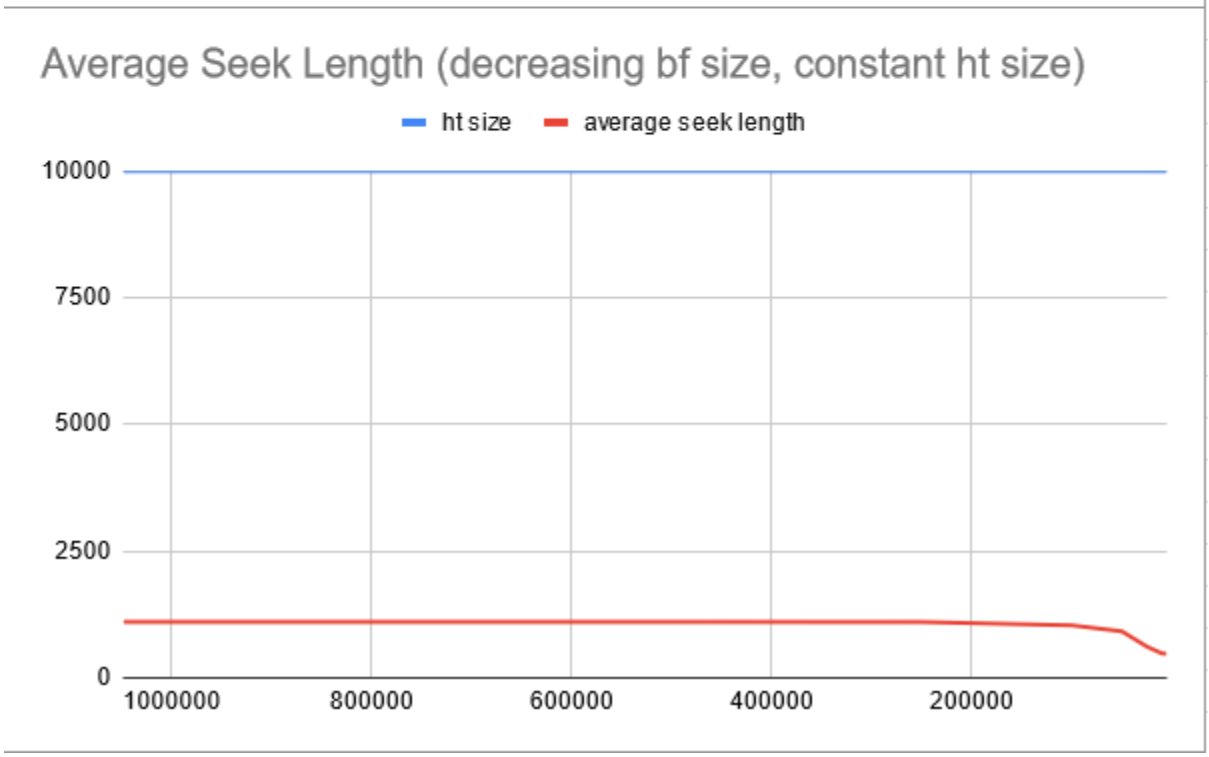
The data I used to create the graphs are below.

HT SIZE	BF SIZE	SEEKS	AVERAGE SEEK LENGTH
10000	1048576	343468	1104
10000	1000000	343468	1104
10000	750000	343468	1104
10000	500000	343498	1104
10000	250000	344114	1102
10000	100000	366106	1036
10000	50000	413082	918
10000	25000	616343	616
10000	10000	788396	482
10000	5000	799603	475

bible.txt

Because I felt that a smaller text file might not provide the desired results, I used the bible.txt file provided by Dr. Long. This is why the number of seeks and average seek lengths are so much larger than the chart prior.

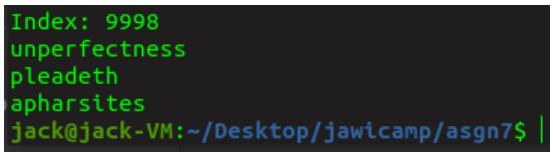

Seeks (decreasing bf size, constant ht size)
bf size
seeks
ht size
1250000
1000000
750000
500000
250000
0
1000000
800000
600000
400000
200000


Average Seek Length (decreasing bf size, constant ht size)
ht size
average seek length
10000
7500
5000
2500
0
1000000
800000
600000
400000
200000

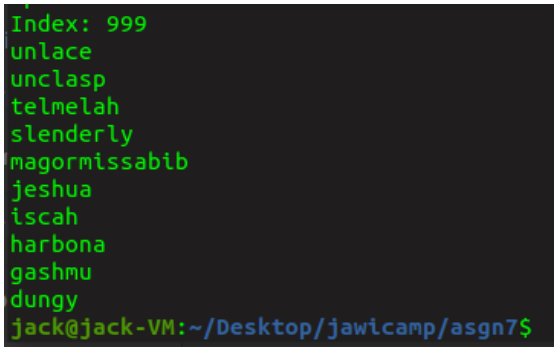
QUESTIONS

Do linked lists get longer?

A - Linked lists get longer as the hash table size decreases. This is because there is less space to fit the same amount of input data. This results in longer linked lists since there isn't enough room for each string to have its own. This is shown in the images below.



The hash table size was set to 10000 in this picture, and contained linked lists of length 1-3.

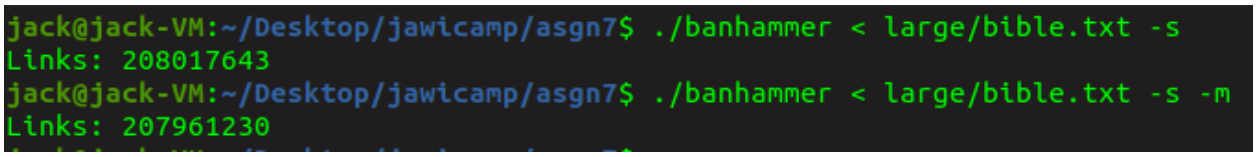


In contrast, when the hash table size is decreased, 1000 in this case, the linked lists increase in length. Another example of this is shown below with a hash table size of 500.

In both cases, when hash table size decreases, the average linked list size increases.

How does the number of links followed without using the move-to-front rule compare to the number of links followed using the rule?

A - Although I believed that the number of links followed would dramatically decrease when the move-to-front rule was applied, there was only a small change. The reasons for this are unbeknownst to me as I've checked my MTF logic as well as how I am incrementing both "seeks" and "links". In the image below, one could see that the amount of links traversed was in fact less when mtf was true, but I believe that my implementation was not optimal. With only a 56,412 link difference between mtf and non-mtf, it is clear I was not able to utilize the logic to its full potential.



How does changing the Bloom filter size affect the number of lookups performed in the hash table?

A - From my testing, decreasing bloom filter size seems to have an exponential relationship with the number of lookups performed. The picture below shows how I came up with this hypothesis.

As the bloom filter decreases from ~1,000,000 to 7500, the number of lookups required initially changes by small amounts and then rapidly increases in size.