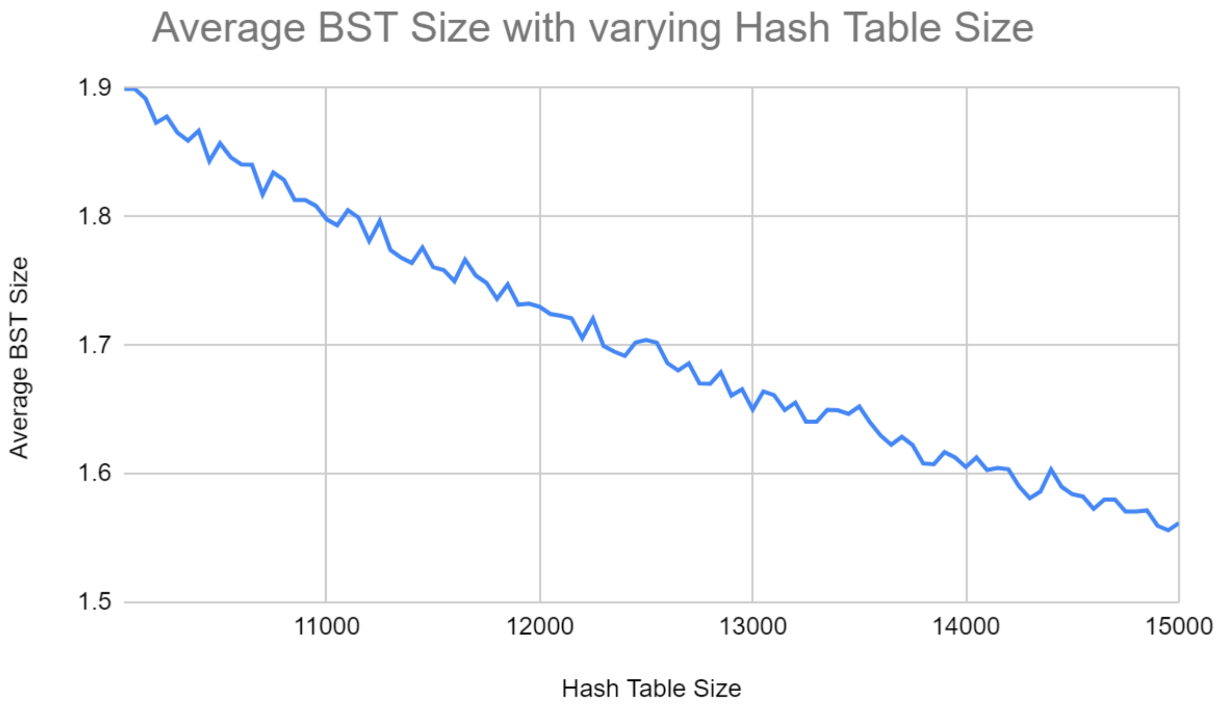
Edison Chen

Assignment 4 - Comparing Varying Binary Search Tree Sizes and Hash Table Sizes

In this assignment, students are required to implement a bloom filter and a hash table to filter out *oldspeak* words, or words you deem improper as the one and only leader of the Glorious People's Republic of Santa Cruz (GPRSC). Statistics such as the average binary tree size, the average binary tree height, average branches traversed, hash table load, and bloom filter load are printed out in the program. This write up will compare the differences between the statistics average binary tree size, average binary tree height, and average branches traversed as the size of the bloom filter and hash table are varied.



Looking at this graph, there is a clear downward trend that can be seen here. As the size of the hash table increases, the average binary search tree size decreases. This downward trend can be attributed to how the hash table is created, in which the array of nodes, *trees* or the binary search tree that is being created along with the hash table is dynamically allocated with the size of the hash table. This causes the number returned from ht_count, which counts the number of non-NULL nodes in the binary search tree, to be much larger. And when the function ht_avg_bst_size is called to calculate the average size of the binary tree, the total sum of the binary search trees within the hash table are divided by the much larger number returned from ht_count. Therefore the downward trend of this graph can be attributed to how the hash table is created and how the array of nodes, or the binary search tree, within the hash table is dynamically allocated depending on the size given to the hash table. What is interesting to note is the bumpiness of the line, which shows some variability when calculating the average binary search tree size. I am not sure about this, but I believe that the bumpiness is caused by some degree of randomness in how the positions of the hash values in the hash table are changed unpredictably depending on the size given to the hash table.

Another thing that I have noticed is that the heights of the binary tree graph is almost identical as the one showing the sizes, with the same noticeable downward trend.

Some properties that can change the average binary tree heights are the sizes of the hash table, the size of the binary search tree, and if the binary tree is balanced or not, meaning that the difference in its left and right subtree are the same or not more than one.

Another interesting to note is that the size of the bloom filter does not affect the average branches traversed at the slightest, perhaps the number of branches and lookups are proportional to each other?