Kamal Chaya, Song Ge
11.25.2013

Answers to Assignment 6 questions

Q1. Give an example of two words that would hash to the same value using stringHash1() but would not using stringHash2().

Words that hash the same value would be words that are made up of the same letters, such as "ate" and "eat". This occurs in stringHash1() but not stringHash2(), since stringHash1() returns a sum of values of each character in a string.

Q2. Why does the above make stringHash2() superior to stringHash1()?

If we assume that each string can only be composed of ASCII characters, the above shows that stringHash1() would have more collisions than stringHash2(), because in stringHash1(), different strings (words) that consist of the same letters would hash to the same value. stringHash2() resolves this by offsetting each character relative to its position(multiplying its ASCII value by its index + 1), so that there will not be a collision when there are two words with the same letters in different orders.

Q3. When you run your program on the same input file but one run using stringHash1() and on the other run using stringHash2(). Is it possible for your size() function to return different values?

No, the size() function returns the count of the hashMap (the amount of hashlinks that the table has). The amount of hashlinks that the table has does not depend on which hash function is used; rather it would be the same in either case.

Q4. When you run your program on the same input file using stringHash1() on one run and using stringHash2() on another, is it possible for your tableLoad() function to return different values?

Load Factor is equal to number of hashLinks divided by the size of the table. The choice of hash function does not effect the amount of hashLinks or the size of the table, so it is not possible for tableLoad() to return different values.

Q5. When you run your program on the same input file with one run using stringHash1() and the other run using stringHash2(), is it possible for your emptyBuckets() function to return different values?

Yes, because different hashFunctions will make words hash to different values, and so the words would be stored in different buckets. This can result in differing amounts of empty buckets.

Q6. Is there any difference in the number of 'empty buckets' when you change the table size from an even number, like 1000 to a prime like 997 ?

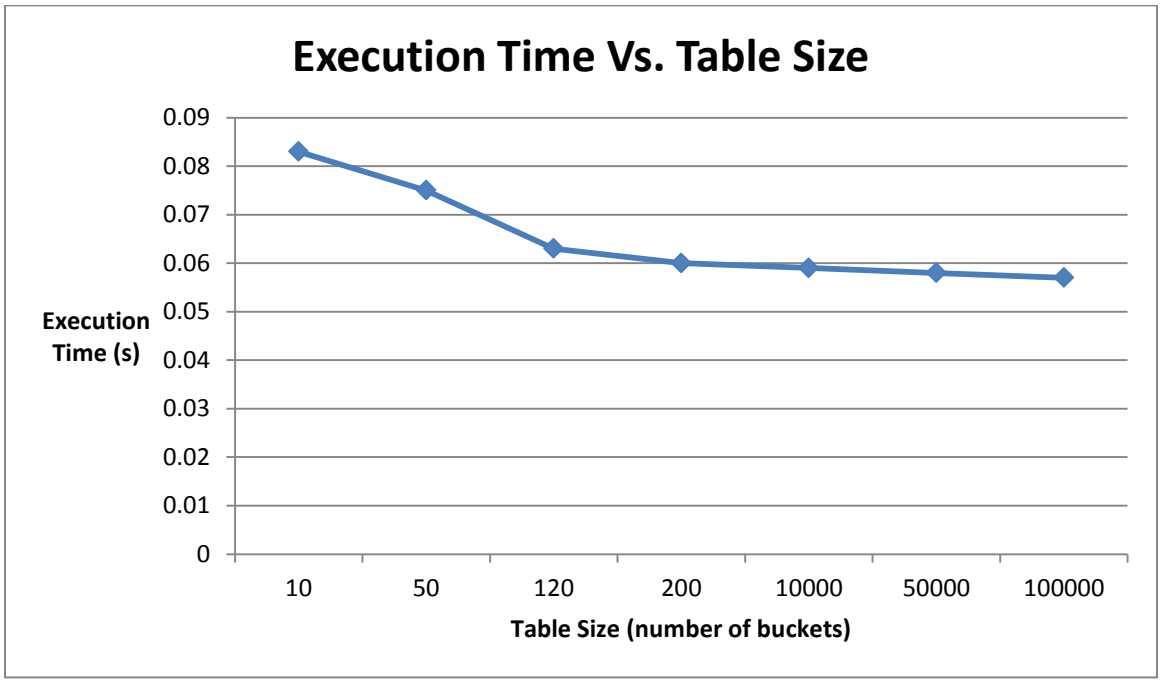
Yes, because the value a word hashes to is dependent on the table size (it is used with the modulus operator and the returned value of the hash function). If the table size is a prime number, this reduces the possibility of common factors from the modulus operator which would mean that words hash to the same bucket.

Q7. Using the timing code provided to you. Run you code on different size hash tables. How does changing the hash table size affect your performance? Please show results as a graph for various table sizes.

A lower number of buckets results in lower performance (longer execution times). This is because if the table size is reduced (meaning the number of buckets is reduced), then that means at each bucket there will be a greater number of hashLinks, which means that the time for searching would be closer to linear time than constant time. Tabulated data is shown below with a graph on the next page:

Table Size	Time Taken (seconds)
100000	0.056
50000	0.058
10000	0.059
200	0.060
120	0.061
50	0.076
10	0.079

Kamal Chaya, Song Ge
11.25.2013



As can be seen on the graph, a smaller table size results in longer execution times, due to the aforementioned reasons. Also, it can be seen how a smaller table size results in execution times that are closer to linear time, since the line of the graph starts to slant as the table size decreases, and how a bigger table size results in execution times closer to constant time (since the graph's line gets closer to a horizontal line).