CS2040S: Data Structures and Algorithms

Tutorial Problems for Week 6: Hashing

For: 12 September 2022, Tutorial 4

Problem 1. Simulation?

In this question we will simulate the operations `add(key)` and `remove(key)` on a `java.util.HashSet`, denoted by the shorthand `I(k)` and `D(k)` respectively. Note that a `HashMap` works on a `<Key, Value>` pair, while in a `HashSet`, the value is the key itself.

Fill the contents of the hash table after each `add` or `remove` operation.
The Hash Table has "table size" of 5, i.e. 5 buckets. The hash function is `h(key) = key % 5`.

Use linear probing as the collision resolution technique:

	0	1	2	3	4
`I(7)`					
`I(12)`					
`I(22)`					
`D(12)`					
`I(8)`					

Use quadratic probing as the collision resolution technique:

	0	1	2	3	4
`I(7)`					
`I(12)`					
`I(22)`					
`I(2)`					

Use double hashing as the collision resolution technique, `g(key) = key % 3`:

	0	1	2	3	4
`I(7)`					
`I(22)`					
`I(12)`					

Use double hashing as the collision resolution technique, `g(key) = 7 - (key % 7)`.

	0	1	2	3	4
`I(7)`					
`I(12)`					
`I(22)`					
`I(2)`					

Problem 2. Hash Functions

A good hash function is essential for good hash table performance. A good hash function is easy/-efficient to compute and will evenly distribute the possible keys. Comment on the flaw (if any) of the following hash functions. Assume the load factor $\alpha = \frac{\text{number of keys}}{\text{table size}} = 0.3$ for all the following cases.

a) The hash table has size 100 with positive even integer keys. The hash function is `h(key) = key % 100`.

b) The hash table has size 49 with positive integer keys. The hash function is `h(key) = (key*7) % 49`.

c) The hash table has size 100 with non-negative integer keys in the range [0, 10000]. The hash function is `h(key)` $= \lfloor \sqrt{\texttt{key}} \rfloor$ `% 100`.

d) The hash table has size 1009, and keys are valid email addresses. The hash function is `h(key) = (sum of ASCII values of each of the last 10 characters) % 1009`. See `http://www.asciitable.com` for ASCII values.

e) The hash table has size 101 with integer keys in the range of [0, 1000]. The hash function is `h(key)` $= \lfloor$ `key` $*$ `random` $\rfloor$ % 101, where `0.0` $\leq$ `random` $\leq$ `1.0`.

f) The hash table has size 54 with `String` keys, with the hash function

```
int hash(String key) {
    h = 0
    for (int i = 0; i <= key.length()-1; i++)
        h += 9 * (int) key.charAt(i)
    h = (h mod 54)
    return h
}
```

Problem 3. String Matching

Text search is a problem where given a long string, the text, you are to find a list of k-letter words hidden in the text. For example, the text "thequickbrownfoxjumpsoverthelazydog" contains the words ("quick", "overt") within it.

Design and implement an algorithm that performs a preprocessing step on the text, so that you can subsequently query the number of occurrences of a k-letter word within the text of length n in $O(k)$ average time. State, with justification, the time complexity of the algorithm.

Problem 4. Finding Sum

You are dining at the hottest new restaurant in town: Algorithms Cafe. As you walk in the door, a bell goes off, and the owner of the restaurant comes out to announce that you are the 2147483647^{th} customer and have won a special deal.

Their menu consists of four components: Appetizers, Soups, Mains, and Desserts. Each component of the menu contains a long and daunting list of n items. If you can choose one item from each section that add up to 100 dollars, then you can eat for free! Come up with the most efficient algorithm to solve this problem, and state the time complexity.

Problem 5. Bloom Filter

Why is there no delete/remove operation for Bloom Filter?