Problem Set 7

Data Structures and Algorithms, Fall 2019

Due: October 31, in class.

From CLRS

Exercise 11.2-3 (justify your answer), 11.2-6, 11.3-3, 11.3-5, 11.4-1, 11.5-1. Problem 11-1.

Additional Problem One

Many theoretical analysis of hashing assumes *ideal random hash functions*. Ideal randomness means that the hash function is chosen *uniformly* at random from the set of *all* functions from U to $\{0, 1, \ldots, m-1\}$. Intuitively, this means for each new item x, we roll a new m-sided die to determine the hash value $h(x)$.

Now, suppose your boss wants you to find a *perfect* hash function for mapping a known set of n items into a table of size m. A hash function is *perfect* if there are no collisions: each of the n items maps to a different slot in the hash table. Of course, a perfect hash function is only possible if $m \geq n$. (This is a different definition of "perfect" than the one considered in class.) After cursing your algorithms instructor for not teaching you about (this kind of) perfect hashing, you decide to try something simple: repeatedly pick ideal random hash functions until you find one that happens to be perfect.

(a) Suppose you pick an ideal random hash function h. What is the *exact* expected number of collisions, as a function of n (the number of items) and m (the size of the table)?

(b) What is the *exact* probability that a random hash function is perfect?

(c) What is the *exact* expected number of different random hash functions you have to test before you find a perfect hash function?

(d) What is the *exact* probability that none of the first N random hash functions you try is perfect?

(e) How many ideal random hash functions do you have to test to find a perfect hash function *with high probability* (that is, with probability at least $1 - 1/n$)?