Practice Problems: Hashing

Problem 3-1. [CLRS 11.4-4] Suppose that we use double hashing to resolve collisions–that is, the hash function is defined as $h(k,i) = (h_1(k) + i \cdot h_2(k)) \mod m$. Show that if m and $h_2(k)$ have a greatest common divisor $d \geq 1$ for some key k, then an unsuccessful search for key k examines $(1/d)$th of the hash table before returning to slot $h_1(k)$.

Problem 3-2. [CLRS 11-4] Let $\mathcal{H}$ be a class of hash functions in which each hash function $h \in \mathcal{H}$ maps the universe U of keys to $\{0, 1, \ldots, m-1\}$. We say that $\mathcal{H}$ is t-universal if, for any given sequence of t distinct keys, $k_1, k_2, \ldots, k_t$, and for any h chosen uniformly at random from $\mathcal{H}$, the sequence $h(k_1), h(k_2), \ldots, h(k_t)$ is equally likely to be any one of the m^t sequences of length t drawn from $\{0, 1, \ldots, m-1\}$.

a. Show that if $\mathcal{H}$ is a 2-universal family, then, $\mathcal{H}$ is universal.

b. Construct a specific family $\mathcal{H}$ that is universal, but not 2-universal, and justify your answer. Write down the family as a table, with one column per key, and one row per function. Try to make $m, \mathcal{H}$, and U as small as possible.

c. Suppose that the universe U is the set of n-tuples of values drawn from $\mathbb{Z}_p = \{0, 1, \ldots, p-1\}$, where, p is prime. Consider an element $x = (x_0, x_1, \ldots, x_{n-1}) \in U$. For any n-tuple $a = (a_0, a_1, \ldots, a_{n-1}) \in U$, define the hash function h_a by

$$h_a(x) = \left(\sum_{j=0}^{n-1} a_j x_j \right) \mod p \ .$$

Let $\mathcal{H} = \{h_a\}$. Show that $\mathcal{H}$ is universal but not two universal. (*Hint*: Find a key for which all hash functions in $\mathcal{H}$ produce the same value.)

d. Suppose that we modify $\mathcal{H}$ slightly from part (**b**): for any $a \in U$ and for any $b \in \mathbb{Z}_p$, define

$$h_{a,b}(x) = \left(\sum_{j=0}^{n-1} a_j x_j + b \right) \mod p \ .$$

Let $\mathcal{H} = \{h_{a,b} \mid a \in U, b \in \mathbb{Z}_p\}$. Show that $\mathcal{H}$ is 2-universal.

e. *Application to authentication.* Suppose that *Ranbir* and *Katrina* secretly agree on a hash function h from a 2-universal family $\mathcal{H}$ of hash functions, where, each $h \in H$ maps the universe of keys U to $\mathbb{Z}_p$, where, p is prime. Now *Ranbir* sends a message m over the internet to *Katrina* and *authenticates* this message by also sending a tag $t = h(m)$. *Katrina* receives the pair

(m,t) and verifies that indeed $h(m) = t$. If the verification succeeds, then she accepts the message, and otherwise discards it. However, there could be many a snooping *Mr. Mediaman* on the internet who can intercept the message (m,t) and replace it with (m', t') and deliver it to *Katrina*. Suppose that the snooping *Medianman* knows the hash family $\mathcal{H}$ (but not the choice h agreed upon by *Ranbir* and *Katrina*). Show that the probability with which *Mr. Mediaman* may succeed in fooling *Katrina* is at most $1/p$, irrespective of how much computing power the *Mediaman* has.

Problem 3-3. [***Rolling Hash Functions for Pattern Matching***] You are given a large piece of text in the string of characters $T[1 \dots n]$. Given a (significantly shorter) pattern $P[1 \dots m]$, *the problem is to determine whether P occurs in T*, that is, if there exists some shift position $0 \le s \le n - m + 1$ such that $P[j] = T[s + j]$, for each $j = 1, \dots, m$. Assume that each character is drawn from an alphabet of 64 characters (to allow upper-case and lower case characters and digits). We can represent any m-character string $C[0, \dots, m-1]$ uniquely as a (large) integer $N(C[0 \dots m-1])$ as follows:

$$N(C[0 \dots m-1]) = C[0] + C[1] \cdot 2^6 + C[2] \cdot 2^{12} + \dots + C[m-1] \cdot 2^{6 \cdot (m-1)}$$

a. Let h be a hash function that maps m character strings to $\mathbb{Z}_p$ (where p is a large prime). Further suppose that h is perfect, that is, for $x \neq y$, $h(x) \neq h(y)$. Assume that you can calculate the hash value of m-character strings in time $O(m)$. Design an algorithm that given the text string $T[1 \dots n]$ and the pattern string $P[1 \dots m]$, returns all positions in T where P occurs, in worst-case time $O(mn)$.

b. Suppose h is not necessarily perfect. Extend the previous algorithm to return all positions in T where P occurs, in worst-case time $O(mn)$.

c. Fix a prime p and define the hash function

$$h_p(x) = x \mod p \ .$$

This hash function can be used to hash any m-character string $A[i \dots i+m-1]$ by first converting it into an equivalent large number $N(A[i \dots i+m-1])$ (as shown above) and then calculating

$$N(A[i \dots i+m-1]) \mod p \ .$$

Show how to calculate the hash value of the string $A[(i+1) \dots (i+m)]$ in $O(1)$ time if the hash value corresponding to the string $A[i \dots (i+m-1)]$ has already been computed and its value is known.

d. Let p be a prime in the range $[2, cn^d]$ for some positive constant c. Let $h_p(x) = x \mod p$ and let $\mathcal{H}$ be the family of hash functions $\mathcal{H} = \{h_p \mid p \text{ is prime and } 2 \le p \le cn^{24}\}$. Let P be the given pattern string of m characters and let $T[i \dots i+m-1]$ be any m-length substring of T. Suppose $n > m$. Show that

$$\Pr_p\left[h_p(N(P)) = h_p(N(T[i \dots i+m-1]))\right] \le O\left(\frac{\log(cn^d)}{cn^d}\right)$$

holds for an appropriate choice of c. ***Hint:*** You could use the following two number theoretic facts: (1) an integer x has at most $\log x$ prime factors, and, (2) the *Prime Number Theorem*: there are $\Theta(x/\log(x))$ prime numbers in the range $[2, x]$.