CS 170 Homework 15

Due **Monday 5/4/2026, at 10:00 pm (grace period until 11:59pm)**

Study Group

List the names and SIDs of the members in your study group. If you have no collaborators, explicitly write "none".

1 Universal Hashing (Solo Question; 10 points)

Let $\mathcal{H}$ be a family of hash functions in which each $h \in \mathcal{H}$ maps the universe $\mathcal{U}$ of keys to $[m] := \{0, 1, \ldots, m-1\}$. $\mathcal{H}$ is *universal* if for every $x \neq y \in \mathcal{U}$, the chance that $h(x) = h(y)$ when we sample h uniformly at random from $\mathcal{H}$ is at most $1/m$.

Consider the following family of hash functions from $[m] \times [m]$ to $[m]$: Let $h_{a,b}(x_1, x_2) = a \cdot x_1 + b \cdot x_2 \mod m$, and let $\mathcal{H} = \{h_{a,b} | a, b \in [m]\}$.

(a) If m is prime, show that this family is universal.

(b) If m is composite, show that this family is not universal.

2 Super-Universal Hashing (10 points)

Let $\mathcal{H}$ be a class of hash functions in which each $h \in \mathcal{H}$ maps the universe $\mathcal{U}$ of keys to $[m] := \{0, 1, \ldots, m-1\}$. Recall the definition of universal $\mathcal{H}$ from Question 1.

We say that $\mathcal{H}$ is super-universal if for every fixed $x \neq y \in \mathcal{U}$, the pair $(h(x), h(y))$ is equally likely to be any of the m^2 pairs of elements of $[m]$ when h is sampled uniformly at random from $\mathcal{H}$. (The probability is taken only over the random choice of the hash function.)

(a) Show that if $\mathcal{H}$ is super-universal, then it is universal.

(b) Suppose that you choose a hash function $h \in \mathcal{H}$ uniformly at random. Your friend, who knows $\mathcal{H}$ but does not know which hash function you picked, tells you a key x, and you tell her $h(x)$. Can your friend then find some $y \neq x$ such that $h(x) = h(y)$ with probability greater than $1/m$ (over your choice of h) if:

(i) $\mathcal{H}$ is universal?

(ii) $\mathcal{H}$ is super-universal?

In each case, either give a choice of $\mathcal{H}$ which allows your friend to find a collision, or prove that they cannot for any choice of $\mathcal{H}$.

3 Streaming Integers (10 points)

In this problem, we are given an infinite stream of positive integers $x_1, x_2, \ldots$, and we have to perform some computation after each new integer is given. Since we may see many integers, we want to limit the amount of memory we have to use in total. For all of the parts below, give a brief description of your algorithm and a brief justification of its correctness.

(a) Show that using only a single bit of memory, we can compute whether the sum of all integers seen so far is even or odd.

(b) Show that we can compute whether the sum of all integers seen so far is divisible by some fixed integer N using $O(\log N)$ bits of memory.

(c) Assume N is prime. Give an algorithm to check if N divides the product of all integers seen so far, using as few bits of memory as possible.

(d) Now let N be an arbitrary integer, and suppose we are given its prime factorization: $N = p_1^{k_1} p_2^{k_2} \ldots p_r^{k_r}$. Give an algorithm to check whether N divides the product of all integers seen so far, using as few bits of memory as possible. Write down the number of bits your algorithm uses in terms of $k_1, \ldots, k_r$.

4 Random Elements of Streams (10 points)

(a) Design an algorithm that takes in a stream $z_1, \ldots, z_T$ of T integers in $[m]$ and at any time t can output a uniformly random element in $z_1, \ldots, z_t$. Your algorithm may use at most polynomial in $\log m$ and $\log T$ space. Prove the correctness and analyze the space complexity of your algorithm. Your algorithm may only take a single pass of the stream.

Hint: what is the probability that the randomly chosen element in $z_1, \ldots, z_t$ is z_t?

(b) For a stream $z_1, \ldots, z_{2m}$ of $2m$ integers in $[m]$, we call $y \in [m]$ a *duplicate element* if it occurs more than once.

Design an algorithm that takes in the stream $z_1, \ldots, z_{2m}$, and with probability at least $1 - \frac{1}{m}$ outputs a duplicate element. Your algorithm may use at most polynomial in $\log m$ space. Prove the correctness and analyze the space complexity of your algorithm. Your algorithm may only take a single pass of the stream.

5 Streaming Majority (10 points)

Design a deterministic algorithm that takes in a stream $z_1, \ldots, z_m$ of integers in $[m]$, and outputs the most common value y in the stream, assuming y occurs more than $m/2$ times. If y occurs at most $m/2$ times, your algorithm may output anything. Your algorithm should use at most polynomial in $\log m$ space.