1. By Chaoren Liu

Question 1

I

To prove the universality, we assume $(x_1, x_2) \neq (x_1^{'}, x_2^{'})$ and the number of functions h_{a_1,a_2} that hashes (x_1, x_2) and $(x_1^{'}, x_2^{'})$ to the same value is at most m.

Without losing the generality, we assume $x_1 \neq x_1^{'}$, and $h_{a_1,a_2}(x_1, x_2) = h_{a_1,a_2}(x_1^{'}, x_2^{'})$, then

$$a_1x_1 + a_2x_2 \equiv a_1x_1^{'} + a_2x_2^{'} \pmod{\text{m}}$$

$$a_1(x_1 - x_1^{'}) \equiv a_2(x_2^{'} - x_2) \pmod{\text{m}}$$

$$a_1 = \frac{a_2(x_2^{'} - x_2)}{x_1 - x_1^{'}} \quad \text{(mod m) because m is prime}$$

For each value of a_2, we have one a_1 and we have m possible a_2. Therefore, this family of hashing functions is universal.

To choose a function, we need to choose $a_1, a_2 \in$ [m] uniformly-independently at random. So we need $2\lceil log_2 m \rceil$ random bits.

II

If m is a fixed power of 2, then we have

$$a_1(x_1 - x_1^{'}) \equiv a_2(x_2^{'} - x_2) \pmod{\text{m}}$$

$$a_1 = \frac{a_2(x_2^{'} - x_2)}{x_1 - x_1^{'}} \pmod{\frac{m}{(m, x_1 - x_1^{'})}}$$

where $(m, x_1 - x_1^{'})$ is the greatest common divisor.

Next we assume $x_1 - x_1^{'} = 2$, then $a_1 = \frac{a_2(x_2^{'} - x_2)}{x_1 - x_1^{'}} \pmod{\frac{m}{2}}$. Because the range of a_1 is [m], there is 2 possible a_1 for each fixed a_2. And a_2 can be any value in [m]. The total number of h_{a_1,a_2} that $h_{a_1,a_2}(x_1, x_2) = h_{a_1,a_2}(x_1^{'}, x_2^{'})$ is more than m. Thereby this family of hash functions is not universal. To choose a function, we need choose a_1 and a_2 uniformly-independently at random from [m]. So we need $2log_2 m$ random bits.

III

Because H is the set of all functions f: [m]→[m-1], the number of functions in H ($|H|$) is $(m-1)^m$. For any pair of $x, y \in [m]$, the number of function f that satisfies f(x)=f(y) is equal to $(m-1)^{m-2}(m-1)$ because f(x)=f(y)=c and c can be any value in [m-1]. Because $(m-1)^{m-2}(m-1) = (m-1)^{m-1} = \frac{|H|}{m-1}$, this set of function is universal. To choose a function independently-uniformly at random we need $\lceil mlog_2(m-1) \rceil$ random bits.

2. By Rupert Freeman

2. (a) Need to choose u to minimize the function

$$(|x_1 - u| + |x_n - u|) + (|x_2 - u| + |x_{n-1} - u|) + ... + (|x_{\frac{n-1}{2}} - u| + |x_{\frac{n+3}{2}} - u|) + (|x_{\frac{n+1}{2}} - u|)$$

Assume that $x_1 \leq x_2 \leq ... \leq x_n$. Note that to minimise $(|x_j - u| + |x_k - u|)$ all we need to do is choose $x_j \leq u \leq x_k$. So to minimise all but the last bracketed term in the function we can choose any $x_{\frac{n-1}{2}} \leq u \leq x_{\frac{n+3}{2}}$. This leaves only the last term to minimise which we do by choosing $u = x_{\frac{n+1}{2}}$, the median.

(b) Need to choose u to minimize the function

$$\begin{aligned}\sum_i (x_i - u)^2 &= \sum_i (x_i^2 - 2x_i u + u^2) \\ &= \sum_i x_i^2 + \sum_i u^2 - \sum_i 2x_i u\end{aligned}$$

Since $\sum_i x_i^2$ is fixed, we need to minimize $f = \sum_i u^2 - \sum_i 2x_i u$. Letting $\sum_i x_i = s$, we get $f = nu^2 - 2us$. Differentiating,

$$\frac{df}{du} = 2nu - 2s$$

and setting to zero to find the critical point,

$$\begin{aligned}2nu - 2s &= 0 \\ u &= \frac{2s}{2n} \\ &= \frac{s}{n} \\ &= \frac{\sum_i x_i}{n}\end{aligned}$$

And by the second derivative test, the critical point is a minimum since $\frac{d^2 f}{du^2} = 2n > 0$.

3. By Chaoren Liu

Question 3

When applying an operation of increment, the number of operations are the number of flippings. We pay two dollars when flipping one bit from 0 to 1, with one dollar paying for the flipping and another one dollar putting on the bit. This extra dollar will be charged when flipping it from 1 to 0. In this case, all bits which are 1 have an extra dollar on it. In this way, reset operation will not be charged. From the algorithm, we can notice that each

increment operation only flip one bit from 0 to 1. Therefore any sequence of n operations take less than 2n flippings, in turn O(n) running time.

By Rupert Freeman

3. Can use the amortized analysis accounting method. Charge 2 units of time for each time a bit is flipped to 1, and 0 units of time each time a bit is flipped to 0. Since the counter starts from zero, this gives an upper bound on the actual time cost. In each increment of the counter only 1 bit is flipped to 1 and in each reset no bits are flipped to 1. Thus the amortized cost of n operations is no greater than $2n$, hence the actual cost is $O(n)$.