Exercise 1 (Frequency Moments and Tug-Of-War) 3+4+1+3+1+2=14 points

Consider the following stream of data elements over the universe $\mathbb{U} = \{1, 2, \ldots, 9\}$:

$$\mathbf{a} = 1, 4, 4, 1, 5, 3, 8, 2, 2, 1, 5.$$

a) Compute $F_0(\mathbf{a})$, $F_1(\mathbf{a})$ and $F_2(\mathbf{a})$.

b) Assume we run the AMS-Estimator (slide 8.36) on $\mathbf{a}$ and evaluate the variables at the end of the while loop (after line 8). Complete the following table with suitable values:

i	1	2	3	4	5	6	7	8	9	10	11
a_i	1			1							
a	1										2
r	1		2				2				2

c) What is the estimated result for $F_2(\mathbf{a})$ returned by the AMS-Estimator in b)?

d) What is the result x^2 returned by the Tug-of-War estimator on $\mathbf{a}$ if the randomly chosen hash function is given by

u	1	2	3	4	5	6	7	8	9
$h(u)$	1	1	-1	1	-1	1	1	1	-1

e) Is there a hash function $h'\colon \mathbb{U} \to \{-1, 1\}$ such that Tug-of-War returns a better (i. e. closer) estimate for F_2 than it does in part d)? If yes, give such a hash function. If no, argue why not.

f) Now think of any stream b with n elements, of which m are distinct. What are the minimum and maximum possible values of $F_2(b)$ (as a function of m and n).

Solution:

a) We have $F_0(\mathbf{a}) = 6$, $F_1(\mathbf{a}) = 11$ and $F_2(\mathbf{a}) = 9+4+1+4+4+0+0+1+0+0 = 23$.

b) The second row a_i is given by the stream.

i	1	2	3	4	5	6	7	8	9	10	11
a_i	1	4	4	1	5	3	8	2	2	1	5
a	1										2
r	1		2				2				2

The remaining values are enforced by:

- having to choose $a = 2$ at $i = 8$ (otherwise we can't have $a = 2$ and $r = 2$ at $i = 11$).

- having to choose $a = 4$ at $i = 2$ (otherwise we can't have $r = 2$ at $i = 3$)
- having to not change $a = 4$ until $i = 8$ (otherwise we can't have $r = 2$ at $i = 7$)

This is the resulting table:

i	1	2	3	4	5	6	7	8	9	10	11
a_i	1	4	4	1	5	3	8	2	2	1	5
a	1	4	4	4	4	4	4	2	2	2	2
r	1	1	2	2	2	2	2	1	2	2	2

c) We choose $k = 2$ and return $11 \cdot (2^2 - 1^2) = 33$.

d) We first need to compute $x = \sum_{i\in[11]} h(a_i) = 1+1+1+1-1-1+1+1+1+1-1 = 5$. The the Tug-Of-War estimator return $x^2 = 25$.

e) We return a value of the form x^2 where x is an integer and 25 is the closest square number to 23, so there can't be a hash function that returns a closer estimate.

f) For the maximum second moment, we pick $(m-1)$-many elements just once each and the remaining one element $(n - m + 1)$ times. This gives us the function

$$1 \cdot (n - m + 1)^2 + (m - 1) \cdot 1^2.$$

For the minimum second moment, we want to distribute the elements as evenly as possible: $(n \ mod \ m)$-many elements of size $\lceil n/m \rceil$, and the remaining elements of size $\lfloor n/m \rfloor$ (i.e. one smaller in size). This gives us the function

$$(n \ mod \ m) \cdot \lceil n/m \rceil^2 + (m - n \ mod \ m) \cdot \lfloor n/m \rfloor^2.$$

Exercise 2 (Improve the Probability) **6 points**

Consider an algorithm $\mathfrak{A}(h)$ that uses a (truly) random hash function $h \in \mathcal{H}$ and gives an estimate $\hat{x} = \mathfrak{A}(h)$ of the true value x of some variable. Suppose that:

$$\Pr_{h \in \mathcal{H}}(\frac{x}{4} \leq \hat{x} \leq 4x) \geq 0.6.$$

The probability of the estimate is with respect to choice of the hash function. How would you compute an estimate x' that has an improved probability of:

$$\Pr(\frac{x}{4} \leq x' \leq 4x) \geq 0.8?$$

Hint: Since we do not know the variance, taking the **average** of multiple runs may not help.

Solution:

Let us choose some k and run the algorithm $k' := 2k - 1$ times with k' different random hash functions that are drawn independently. Let $\hat{x}_1, ..., \hat{x}'_k$ be the sorted resulting estimations. Return the median $x' := \hat{x}_k$ as the new estimation.

We can follow the proof for the Approximation Guarantee on Page 8.31 to find that $58.06 \leq k$.

Alternatively we can calculate a simple and better bound by hand:

Now let us assume that the new estimate x' is not within the range $\frac{x}{4} \leq x' \leq 4x$. This means we must have sampled a hash function h such that $\hat{x}$ is outside of this range at least $\frac{k'+1}{2}$ times (otherwise the median is in the range).

Conversely, if we sample outside of the range $i < \frac{k'+1}{2}$ times, then the returned estimator x' is clearly also within the range $\frac{x}{4} \leq x' \leq 4x$. The probability for this is at least:

$$\Pr(\frac{x}{4} \leq x' \leq 4x) \geq \sum_{i=0}^{\frac{k'-1}{2}} \binom{k'}{i} \cdot 0.4^i \cdot 0.6^{(k'-i)}$$

For $k' = 17$ or $k = 9$ we have $\Pr(\frac{x}{4} \leq x' \leq 4x) \geq 0.801$.

Exercise 3 (Minimum Memory for Distinct Elements Approximation) 0 points

Show that any deterministic algorithm that even guarantees to **approximate** the number of distinct elements in a data stream over universe $\mathbb{U} = \{1, ..., m\}$ with error less than $\frac{m}{16}$ must use $\Omega(m)$ bits of memory. This is even the case for data streams of length less than $2m$.

Hint: There is a constant $c > 0$, for which it is possible to create 2^{cm} subsets of $\{1, ..., m\}$, each with $m/2$ elements, such that no two of the subsets have more than $3m/8$ elements in common. You can use this fact without proving it.

Solution:

Let $\mathfrak{S}$ be the set of 2^{cm} subsets of $\{1, ..., m\}$, each with $m/2$ elements, such that no two of the subsets have more than $3m/8$ elements in common (as given by the hint). We denote the sets by $S_i \in \mathfrak{S}$ and we define the stream s_i as a stream containing every element in S_i exactly once in ascending order.

We now show that any algorithm with a guaranteed error of less than $\frac{m}{16}$ needs to use at least 2^{cm} distinct states and therefore memory of at least $c \cdot m$ bits.

Assume we use less than $c \cdot m$ bits. Then there are two sets $S_1, S_2 \in \mathfrak{S}$, such that after reading the streams s_1 and s_2 we end in the same state. Now consider appending these streams by s_1. Then the resulting streams $s_1 \cdot s_1$ and $s_2 \cdot s_1$ end in the same state and we return the same result y for both streams. By definition of the sets, $s_1 s_1$ has exactly $y_1 = m/2$ distinct elements and $s_2 s_1$ has at least $y_2 \geq m/2 + m/8$ distinct elements. This is because the two sets S_1 and S_2 differ in at least $m/8$ elements. The streams have a length of $2 \cdot m/2 = m$ elements.

Now assume the errors are $|y_1 - y| < m/16$ and $|y_2 - y| < m/16$, then $|y_1 - y_2| < m/8$. We know from the construction of the strings that $|y_1 - y_2| \geq m/8$, which leads to a contradiction.

Therefore we show that the algorithm must be able to distinguish between at least 2^{cm} different states (representing the 2^{cm} sets) and therefore must use at least $\log(2^{cm}) = cm$ bits, which can be written as $\Omega(m)$ bits.