Exercise Set VIII, Advanced Algorithms 2022

These exercises are for your own benefit. Feel free to collaborate and share your answers with other students. Solve as many problems as you can and ask for help if you get stuck for too long. Problems marked * are more difficult but also more fun :).

These problems are taken from various sources at EPFL and on the Internet, too numerous to cite individually.

1 Professor Ueli von Gruyères has worked intensely throughout his career to get a good estimator of the yearly consumption of cheese in Switzerland. Recently, he had a true breakthrough. He was able to design an incredibly efficient randomized algorithm $\mathcal{A}$ that outputs a random value X satisfying

$$\mathbb{E}[X] = c \qquad \text{and} \qquad \mathrm{Var}[X] = c^2 \,,$$

where c is the (unknown) yearly consumption of cheese in Switzerland. In other words, $\mathcal{A}$ is an unbiased estimator of c with variance c^2.

Use Ueli von Gruyères' algorithm $\mathcal{A}$ to design an algorithm that outputs a random value Y with the following guarantee:

$$\Pr[|Y - c| \geq \epsilon c] \leq \delta \qquad \text{where } \epsilon > 0 \text{ and } \delta > 0 \text{ are small constants.} \tag{1}$$

Your algorithm should increase the resource requirements (its running time and space usage) by at most a factor $O(1/\epsilon^2 \cdot \log(1/\delta))$ compared to the requirements of $\mathcal{A}$.

(In this problem you are asked to (i) design the algorithm using $\mathcal{A}$, (ii) show that it satisfies the guarantee (1)*, and (iii) analyze how much the resource requirements increase compared to that of simply running $\mathcal{A}$. Recall that you are allowed to refer to material covered in the course.)*

Solution: The idea of the algorithm is to first decrease the variance by taking the average of $t = 10/\epsilon^2$ independent runs of $\mathcal{A}$. We then do the median trick. Formally, consider the algorithm $\mathcal{B}$ that runs t independent copies of $\mathcal{A}$ and then outputs the average of the t estimates obtained from the independent runs of $\mathcal{A}$. Let B be the random output of this algorithm. As seen in class, we have $\mathbb{E}[B] = c$ (it is still an unbiased estimator) and $\mathrm{Var}[B] = c^2/t$. Now by Chebychev's Inequality we have

$$\Pr[|B - c| \geq \epsilon c] \leq \frac{\mathrm{Var}[B]}{\epsilon^2 c^2} = \frac{1}{t\epsilon^2} = 1/10 \qquad \text{(since we selected } t = 10/\epsilon^2\text{)}\,.$$

So algorithm $\mathcal{B}$ returns a $1 \pm \epsilon$ approximation with probability at least $9/10$. We now want to decrease the probability $1/10$ of failing all the way down to δ. To do this we use the median trick. Let $\mathcal{C}$ be the algorithm that runs $u = 10\ln(1/\delta)$ independent copies of $\mathcal{B}$ and outputs the *median* of the obtained copies. Let Y be the random output of $\mathcal{C}$. We now analyze the failure probability of $\mathcal{C}$, i.e., we wish to show $\Pr[|Y - c| \geq \epsilon c] \leq \delta$. To do so define $Z_i \in \{0, 1\}$ to be the indicator random variable that takes value 1 if the i:th run of $\mathcal{B}$ outputs a value B_i such that $|B_i - c| \geq \epsilon c$. Note that $\Pr[|B_i - c| \geq \epsilon c] \leq 1/10$ and so $\Pr[Z_i = 1] \leq 1/10$. So if we let $Z = Z_1 + Z_2 + \cdots + Z_u$, then Z is a sum of independent variables where $\mathbb{E}[Z] \leq u/10$. Moreover since Y is the median of the independent runs of $\mathcal{B}$,

$$\Pr[|Y - c| \geq \epsilon c] \leq \Pr[Z \geq u/2] .$$

We shall now analyze $\Pr[Z \geq u/2]$ using the Chernoff Bounds. Indeed, since Z is a sum of *independent* random variables taking values in $\{0, 1\}$ we have

$$\Pr[Z \geq u/2] \leq \Pr[Z > 3 \cdot \mathbb{E}[Z]] \leq e^{-\ln(1/\delta)} = \delta .$$

We have thus proved that $\mathcal{C}$ satisfies the right guarantees. Let us know analyze its resource requirements. $\mathcal{C}$ runs $O(\log(1/\delta)$ copies of $\mathcal{B}$ and each copy of $\mathcal{B}$ runs $O(1/\epsilon^2)$ copies $\mathcal{A}$. Thus the total resource requirements increase by at most a factor $O(\log(1/\delta)1/\epsilon^2)$ as required (calculating the mean and median can be done in linear time so it does not affect the asymptotic running time).

2 Suppose that you are given an insertion only stream of items. For every $k \geq 1$, give an algorithm that at each point in the stream maintains k uniformly random elements from the prefix of the stream sampled without replacement. Your algorithm must use $O(k \log n)$ space.

Solution: This is known as *reservoir sampling*. The algorithm is as follows:

1. Keep the first k items in memory.
2. When the i-th item arrives (for $i > k$)
   - with probability k/i, keep the new item and discard a uniformly random item of those that are currently in memory;
   - with probability $1 - k/i$, keep the old items and ignore the new one.

We will perform an induction on the number of elements m that the maintained set is a uniformly random set of k items from the stream. If $m \leq k$, the algorithm is clearly correct: this provides the base of the induction. Let us assume that till some time step $j - 1$, the maintained set is a uniformly random subset of the first $j - 1$ elements of size k.

The inductive step is provided by the following argument. Note that the probability that the j-th element that arrives in the stream belongs to the set of k uniformly random elements from $1, \ldots, j$ sampled without replacement is exactly

$$\binom{j-1}{k-1} / \binom{j}{k} = \frac{(j-1)!}{(k-1-(j-1))!(k-1)!} \cdot \frac{(k-j)!k!}{j!} = \frac{k}{j}.$$

If j is included, then it suffices to add to $\{j\}$ a uniformly random subset of the first $j - 1$ elements of size $k - 1$. Taking a uniformly random element out of the maintained set of size k achieves exactly this goal.

3 Consider a data stream $\sigma = (a_1, \ldots, a_m)$, with $a_j \in [n]$ for every $j = 1, \ldots, m$, where we let $[n] := \{1, 2, \ldots, n\}$ to simplify notation. For $i \in [n]$ let f_i denote the number of times element i appeared in the stream σ.

We say that a stream σ is *approximately sparse* if there exists $i^* \in [n]$ such that $f_{i^*} = \lceil n^{1/4} \rceil$ and for all $i \in [n] \setminus \{i^*\}$ one has $f_i \leq 10$. We call i^* the *dominant* element of σ. Give a single pass streaming algorithm that finds the dominant element i^* in the input stream as long as the stream is approximately sparse. Your algorithm should succeed with probability at least $9/10$ and use $O(n^{1/2}\log^2 n)$ bits of space. You may assume knowledge of n (and that n is larger than an absolute constant).

Solution: In class that we showed that using AMS sketch we can approximate the L_2 norm of a vector within a factor ϵ with constant probability by maintaining a vector with $O(\frac{1}{\epsilon^2})$ entries. We can then use the median trick to obtain our estimate with probability of failure at most δ by using a vector with $O(\frac{1}{\epsilon^2}\log(\frac{1}{\delta}))$ entries, hence requiring $O(\log(\frac{1}{\epsilon^2\delta})\log(n))$ space. We use this observation below.

First, we partition the universe into $\sqrt{n}$ disjoint blocks $[n] = B_1 \cup \ldots \cup B_{\sqrt{n}}$ each of size $\sqrt{n}$. Denote the corresponding frequency vectors by $f^1, \ldots, f^{\sqrt{n}} \in \mathbb{R}^{\sqrt{n}}$. The algorithm is as follows. For every $j \in [\sqrt{n}]$ and every $i \in B_j$ we use the AMS sketch with ϵ a sufficiently small constant (to be shown later) and $\delta = 1/n^2$ to obtain a $(1 \pm \epsilon)$-approximation to

$$||f^j||_2^2$$

and

$$||f^j - \lceil n^{1/4} \rceil \cdot e_i||_2^2.$$

Let a^j be the estimate of $||f^j||_2^2$ and a_i^j be the estimate of $||f^j - \lceil n^{1/4}\rceil \cdot e_i||_2^2$ respectively.

Let $i \in B_j$. Then if we chose ϵ such that

$$(1+\epsilon)||f^j - \lceil n^{1/4}\rceil \cdot e_{i^*}||_2^2 < (1-\epsilon)||f^j||_2^2$$

and

$$(1+\epsilon)||f^j||_2^2 < (1-\epsilon)||f^j - \lceil n^{1/4}\rceil \cdot e_i||_2^2$$

for $i \neq i^*$, then since with probability at least $\frac{1}{n^2}$ each of the following hold

$$(1-\epsilon)||f^j||_2^2 \leq a^j \leq (1+\epsilon)||f^j||_2^2$$

$$||f^j - \lceil n^{1/4}\rceil \cdot e_i||_2^2 \leq a_i^j \leq (1+\epsilon)||f^j - \lceil n^{1/4}\rceil \cdot e_i||_2^2,$$

we can take the union bound over these equations to get that with probability at least $\frac{9}{10}$ all of these equations hold.

All that remains is to show that it suffices to choose a constant size value of ϵ. This follows because

$$||f^j - \lceil n^{1/4}\rceil \cdot e_i||_2^2 - ||f^j||_2^2 = \Omega(1)||f^j||_2$$

if $i \neq i^*$ and

$$||f^j - \lceil n^{1/4}\rceil \cdot e_{i^*}||_2^2 - ||f^j||_2^2 = -\Omega(1)||f^j||_2.$$

In this case we guess i^* to be the element i such that $a_i^j < a^j$.

4 *(12 pts)* **Alice, Bob and Charlie.** Suppose that Alice and Bob have two documents d_A and d_B respectively, and Charlie wants to learn about the difference between them. We represent each document by its word frequency vector as follows. We assume that words in d_A and d_B come from some dictionary of size n, and let $x \in \mathbb{R}^n$ be a vector such that for every word $i \in [n]$[1] the entry x_i equals the number of times the i-th word in the dictionary occurs in d_A. Similarly, let $y \in \mathbb{R}^n$ be a vector such that for every word $i \in [n]$ the entry y_i denotes the number of times the

[1] We let $[n] := \{1, 2, \ldots, n\}$.

i-th word in the dictionary occurs in d_B. We assume that the number of words in each document is bounded by a polynomial in n.

Suppose that there exists $i^* \in [n]$ such that for all $i \in [n] \setminus \{i^*\}$ one has $|x_i - y_i| \leq 2$, and for i^* one has $|x_{i^*} - y_{i^*}| \geq n^{1/2}$. Show that Alice and Bob can each send a $O(\log^2 n)$-bit message to Charlie, from which Charlie can recover the identity of the special word i^*.

Your solution must succeed with probability at least $9/10$. You may assume that Alice, Bob and Charlie have a source of shared random bits.

Solution: Alice and Bob can both apply the AMS sketch with constant precision and failure probability $1/n^2$ to their vectors. Then Charlie subtracts the sketches from each other, obtaining a sketch of the difference. Once the sketch of the difference is available, one can find the special word similarly to the previous problem.