Homework 2

Algorithms for Big Data
CS498ABD Spring 2019
Due: 10am, Friday, Feb 22nd

Instructions:

- Each home work can be done in a group of size at most two. Only one home work needs to be submitted per group. However, we recommend that each of you think about the problems on your own first.
- Homework needs to be submitted in pdf format on Gradescope. See `https://courses.engr.illinois.edu/cs374/fa2018/hw-policies.html` for more detailed instructions on Gradescope submissions.
- Follow academic integrity policies as laid out in student code. You can consult sources but cite all of them including discussions with other class mates. Write in your own words. See the site mentioned in the preceding item for more detailed policies.

Exercise 1: Experimenting with Morris Counters In class we considered the Morris counting algorithm, which initialized a counter X to 1, and for every increment instruction, incremented X with probability $1/2^X$. The estimator was $2^X - 1$.

We also saw two variants of the counter. In class, we took the median of averages of independent copies of the basic Morris counter. In order to obtain a $(1 \pm \epsilon)$-approximation with probability at least $1 - \delta$, we need $O(\frac{1}{\epsilon^2} \log \frac{1}{\delta} \log \log n)$ bits. In the previous homework, instead of incrementing with probability $1/2^X$, we incremented X with probability $1/(1+a)^X$ for some $a > 0$ (and used a different estimator). One can show that this version only needs $O(\log \frac{1}{\epsilon} + \log \frac{1}{\delta} + \log \log n)$ bits to achieve a $(1 \pm \epsilon)$-approximation with probability $1 - \delta$.

Using any language of your choice, implement the following:

- The basic Morris counter from class
- The median-of-averages variant from class
- The variant from the previous homework

Feel free to experiment with different choices of ϵ and δ for the two variants.

Submit the following:

1. A high-level explanation of the choices you made in how you implemented the counters. Also tell us how many parallel copies/what a is for the variants, and the values of ϵ, δ you are shooting for.

2. Give a plot comparing n to the estimator from each of the three counters. Also give a plot comparing n to the number of bits needed to estimate n (you don't actually have to implement bit-counters, feel free to use something like `int` or `long`, just show how many bits you would have used if you implemented a bit counter). Generate your plots by computer; do not hand draw them.

3. How does the experimental performance and space of your implementation compare to the theoretical guarantees? How do you think your implementation details might affect the experimental performance? Feel free to include any other relevant (or surprising) observations.

Exercise 2: Distinct Elements This is mainly to make you work out a simple distinct elements analysis for yourself. Here is a variant of the algorithm we saw in lecture. Instead of using an ideal hash function h we choose a random hash function $h : [n] \to [n]$ from pairwise-independent hash family $\mathcal{H}$. Let Z be the minimum hash value seen in the stream. Suppose the number of distinct elements d is in the range $[2^i, 2^{i+1})$. Prove that $\mathrm{P}[Z \in [n/2^{i+3}, n/2^{i-2})] > c$ for some fixed constant c. Thus n/Z gives a constant factor estimate for d with probability at least c.

Hint: You may find it useful to give a bound on the probability that nothing hashes into $[1, n/2^{i+3})$.

Exercise 3: Sampling from Distinct Elements We saw how to estimate the number of distinct elements from a stream. Now we consider the problem of sampling nearly uniformly from the set of distinct elements. If we had access to an ideal hash function $h : [n] \to [0, 1]$ then one can see that the element which achieves the minimum hash value is uniformly distributed among the distinct elements. However, we do not have access to an ideal hash function. There is a notion of minwise independent hash functions which is directly relevant to this application; we will see their formal definition later in the course. For now we will see how to modify the algorithm we saw in lecture to obtain a near uniform sample from the distinct elements.

Consider the BJKST algorithm we saw in lecture that used a random hash function $h : [n] \to [n^3]$ from a pairwise independent hash family $\mathcal{H}$. Now assume that we instead use a 3-wise independent hash family. The algorithm stores $t = 1/\epsilon^3$ elements associated with the smallest t hash values seen and at the end of the algorithm outputs one of them uniformly at random. Our goal is to show that each distinct element is output with probability at least $(1 - 10\epsilon)/d$ and at most $(1 + \epsilon)/d$ (i.e., nearly uniform). (You may assume that ϵ is smaller than some constant like $1/2$ to make the calculations easier.)

To this end, let $b_1, b_2, \ldots, b_d$ are the distinct values in the stream. Assume $d > 1/\epsilon^3$ for otherwise we can store all of them and output a uniformly sampled element. Observe that an element b_i is among the t remaining elements if each of the following events all occur.

(a) $h(b_i) \leq \lceil (1 - \epsilon)tN/d \rceil$.

(b) The number of other elements b_j $(j \neq i)$ such that $h(b_j) < \lceil (1 - \epsilon)tN/d \rceil$ is at most $t - 1$.

(c) $h(b_j) \neq h(b_i)$ for all $j \neq i$.

Show that the above events all occur with probability close to t/d via the following steps.

1. Let Z be an indicator for $h(b_i) \leq \lceil (1-\epsilon)tN/d \rceil$. What is $\mathrm{P}[Z=1]$?
2. Conditioned on $Z=1$, show that the probability that more than $t-1$ of the remaining items (b_j where $j \neq i$) has value $< \lceil (1-\epsilon)tN/d \rceil$ is at most $(1+\epsilon)\epsilon$.
3. Show that the probability of a hash collision with b_i is at most $\epsilon t/d$.

   Hint: First show the probability is at most $O(1/n^2)$
4. Put the above together to show that b_i is one of the t selected elements with probability $\geq (1-10\epsilon)t/d$, hence output with probability $\geq \dfrac{1-10\epsilon}{d}$.
5. **Extra credit:** Show that the probability that b_i is output is at most $(1+10\epsilon)/d$.

Make note in particular of where you use the assumption that h is 3-wise independent.

Problem 4. In class, we saw how to the AMS sampling procedure allows us to estimate the kth moment F^k in sublinear space for $k \geq 2$. Recall also that the AMS sampler still requires polynomial space because the variance of a single sample was polynomial. Here we will use AMS to estimate the *entropy* of a stream; in particular, we will show that we only need *logarithmic space* (modulo dependencies on ϵ and δ) to estimate the entropy.

Let $f \in \mathbb{Z}^n_{\geq 0}$ be frequency counts over n elements, and for each i, let $p_i = \dfrac{f_i}{m}$ be the corresponding probability distribution. The *entropy* of p is the quantity $\Phi = \displaystyle\sum_i p_i \ln \frac{1}{p_i}$, where $0 \ln\left(\frac{1}{0}\right) = 0$. Our high-level goal is to obtain an $(1 \pm \epsilon)$-multiplicative approximation to the entropy, but there is a technical issue because the entropy can be zero. We instead seek a $(1 \pm \epsilon)$-multiplicative approximation of $1 + \Phi$, which converts to a $(1 \pm 2\epsilon)$-multiplicative approximation of Φ if $\Phi \geq 1$ and a 2ϵ-additive approximation of Φ if $\Phi \leq 1$. You may assume m is larger than a fixed constant, say 42.

Let $g(\ell) = \dfrac{\ell}{em} \ln\left(\dfrac{me}{\ell}\right)$.

1. Show that $1 + \Phi = e \displaystyle\sum_{i \in [n]} g(f_i)$.
2. Show that $g(\ell) \geq g(\ell - 1)$ for $\ell \leq m$.
3. Show that for $\ell \leq m$, $g(\ell) - g(\ell-1) \leq \dfrac{1 + \ln m}{me}$
4. Let Y be the AMS sample for the quantity $\dfrac{1}{e}(1+\Phi) = \displaystyle\sum_i g(f_i)$. We know from class that $\mathrm{E}[Y] = \dfrac{1}{e}(1+\Phi)$. Show that

$$\mathrm{Var}[Y] \leq c_1(1 + \ln(m))(1+\Phi) = ec_1(1+\ln(m))\,\mathrm{E}[Y]$$

for some constant $c_1 > 0$.

5. Let $t = \dfrac{c(1+\ln m)}{\epsilon^2}$ for some (suitable) constant $c > 0$. For $i \in [t]$, let Y_i be an independent instance of the AMS sample for $\sum_i g(f_i)$, and let $Z = \dfrac{1}{t}\sum_{i=1}^{t} Y_i$ be the sum. Show that

$$\mathrm{P}[|eZ - (1+\Phi)| \geq \epsilon(1+\Phi)] \leq \frac{1}{4}$$

for some constant $c > 0$.

6. Finally, explain how to arrange $O\big(\ln(1/\delta)\ln(m)/\epsilon^2\big)$ independent AMS samples to obtain a $(1 \pm \epsilon)$-approximation of $1 + \Phi$ with probability $\geq 1 - \delta$.