Homework 2

Algorithms for Big Data
CS498ABD Spring 2019
Due: 10am, Wednesday, Feb 20th

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 $(1+\epsilon)$-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 $(1+\epsilon)$-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 $[0, n/2^{i+3})$.

Exercise 3,4: To be announced...