6.856 — Randomized Algorithms

David Karger

Apr. 1, 2021 — Problem Set 7, Due 4/7

Problem 1. Basic Sampling Tricks

(a) Amplification for sampling. Suppose you have an estimation algorithm that will find a $(1 \pm \epsilon)$ approximation to the correct value with probability $3/4$. Show that you can reduce the failure probability exponentially fast from $1/4$ to any desired δ by performing some number k of estimation experiments and taking the median value returned. Give the smallest upper bound you can on k as a function of δ. This shows the $\log(1/\delta)$ term in $\mu_{\epsilon,\delta}$ is natural.

(b) Why do we use the median rather than the average of the repeated samples in the previous part?

(c) Error bound for sampling. Suppose you can generate independent samples of a random variable whose standard deviation is less than its mean. Give an (ϵ, δ)-FPRAS for estimating the mean of this distribution (to within $1 \pm \epsilon$ with probability $1-\delta$) with a number of samples polynomial in $1/\epsilon$ and $\log 1/\delta$. **Hint:** Consider the sum of n independent samples from the distribution and determine its mean and variance. Bound the probability that this sum deviates greatly from its mean. Now use the previous parts. This shows the $1/\epsilon^2$ term in $\mu_{\epsilon,\delta}$ is natural.

(d) An *unbiased estimator* for a quantity μ is a random variable X whose expectation is μ. If X has variance σ^2, the *relative variance* for X is defined as σ^2/μ^2. Show that if X has relative variance r then taking $O(r\epsilon^{-2} \ln 1/\delta)$ samples of X will yield an (ϵ, δ)-approximation for μ.

Problem 2. Suppose you are given a directed graph with n vertices and m unit-length edges. Consider the problem of estimating the number of vertices within distance d of each vertex. Give a fully polynomial (ϵ, δ) approximation scheme that solves this problem simultaneously for all vertices for any fixed constant d. Your running time should be $O((m+n)/\epsilon^2 \log 1/\delta))$ to within polylogarithmic factors.

Problem 3. Consider a stream of N elements $x_1, x_2, \cdots, x_N$ from the universe $[m]$. For any such stream, define $h \in \mathbb{Z}_{\geq 0}^m$ as the histogram of samples seen, that is, h_i is the number of times i occurred in the stream. In class, we learnt how to estimate the number of distinct elements seen in a stream of N elements from $[m]$, also represented as $||h||_0$. The goal of this problem is to estimate the second moment of the histogram $||h||_2^2 := \sum_{i=1}^m h_i^2$ (also known as the "repeat rate").

(a) Show that for any vector $h \in \mathbb{Z}_{\geq 0}^m$, and for a random vector $x \in \{+1, -1\}^m$, the quantity $\langle h, x \rangle^2$ has expected value $||h||_2^2$ and standard deviation of $O(||h||_2^2)$.

(b) Assuming that we have access to arbitrary amounts of randomness, show how we can estimate $||h||_2^2$ upto multiplicative $(1 \pm \varepsilon)$ error with probability at least $1 - \delta$, using space $O_{\varepsilon,\delta}(\log N) = O(\varepsilon^{-2} \log(1/\delta) \log N)$ if the space for storing the randomness is not counted.

(c) Show how you can get rid of the unrealisitic assumption of unbounded randomness by using $O(1)$-wise independent randomness, therby truly using only $O_{\varepsilon,\delta}(\log N + \log m)$ space.

Problem 4—This problem should be done without collaboration. In class we talked about how to count the number of satisfying assignments to a DNF, which is equivalent to estimating the probability of a satisfying assignment if variables get unbiased random assignments. Suppose that instead each variable gets set to true with uniform probability $p < 1/2$. Show how to estimate the probability of getting a satisfying assignment.

Problem 5. Based on MR 11.2. In class we gave an FPRAS for DNF counting that evaluated the entire formula about m times, where m is the number of clauses. Here we develop a faster algorithm. Consider the following variant of the DNF counting algorithm from class. For the t-th trial, pick a satisfying assignment a uniformly at random from the *disjoint* union of satisfying assignments, just as described in class. But now, instead of checking whether a is the "first" copy of itself, try the following. Let N be the number of assignments in the disjoint union. Let c_a be the number of clauses that a satisfies. Define $X_t = 1/c_a$.

(a) Prove that $N \cdot E[X_t]$ is the number of satisfying assignments to the DNF formula

(b) Prove that $O(m\mu_{\epsilon\delta})$ trials (and computation of the resulting $\sum X_t$) suffice to DNF-count to within multiplicative error $(1 \pm \epsilon)$ with probability $1 - \delta$. **Hint:** use a Chernoff bound generalization that is not limited to binary random variables.

(c) Once a has been chosen in the previous subproblem, give an algorithm for quickly estimating c_a to within $(1 \pm \epsilon)$. Argue that this is sufficient to give us the $(1 \pm \epsilon)$ approximation for DNF-counting.

(d) Using the new scheme from the previous subproblem, analyze the expected number of clauses you need to evaluate in the course of the algorithm. Assuming all clauses are the same size, what is the actual running time of the scheme in terms of basic operations?

(e) **Optional:** Justify the assumption that all clauses are the same size to within a logarithmic factor, thus extending the runtime analysis to arbitrary formulae.

Problem 6—Optional. Many simulations requiring sampling from the *exponential* distribution ($\Pr(X > t) = e^{-t}$). The easiest method is to generate a random variable uniformly distributed in $[0, 1]$ and take its logarithm, but this requires expensive math. Von Neumann developed a strange and clever trick for generating such samples:

- Set $k = 0$
- Generate a sequence of random variables $u_0, \ldots, u_n$ so long as they decrease, i.e. until $u_{n+1} > u_n$.
- If n is even then output $x = u_1 + k$. Otherwise increment k and go back to the previous step.

Analyze this algorithm.

(a) Prove that its outputs are exponentially distributed. **Hint:** start by determining the distribution of n.

(b) Bound the expected number of (uniform) random samples needed to output one exponential sample.