6.856 — Randomized Algorithms

David Karger

Apr. 14, 2021 — Problem Set 9, Due 4/23

Problem 1. Our in-class analysis of sampling graph edges doesn't work for directed graphs. But it applied to analyze the directed *residual graphs* that emerge during max-flow computations, to yield faster max-flow algorithms. This problem assumes knowledge of the augmenting paths algorithm for max-flow.

Suppose unweighted and undirected graph G has s-t max-flow v and global min-cut c. Suppose you have found an s-t flow of value $f \leq v$ and consider the residual graph of this flow.

(a) Show that an s-t cut of value r in G has (directed) value at least $r - f$ in the residual graph.

(b) Prove that if $\tilde{O}(\frac{mv}{c(v-f)})$ edges of G are selected at random, then with high probability at least one edge (directed from s to t in the residual graph) of each s-t cut will be selected. Conclude that the sample will contain an augmenting path.

(c) Use the previous problem to design a max-flow algorithm with runtime $\tilde{O}(mv/c)$. With additional work, the time can be pushed down to $\tilde{O}(m + nv)$.

This improves on the $\tilde{O}(mv/\sqrt{c})$ bound you proved before.

Problem 2. In class we discussed the relationship between *counting* (estimating a probability) and *generation* (sampling from a distribution). We also gave an (ϵ, δ)-FPRAS for estimating the probability of a graph G disconnecting when each edge fails with probability p. You will show how to sample a random disconnected version of G from this distribution.

Let G be a graph and F the event that it fails. Let x_e be the state of edge e (up or down).

(a) Explain how $\Pr(F \mid x_e)$ can be computed as a network reliability problem on a different graph, for both values of x_e.

(b) Give an FPRAS for computing $\Pr(x_e \mid F)$.

(c) Using self-reducibility, give an algorithm that produces a random disconnected version of G, conditioned on F. The distribution should (ϵ, δ)-approximate the distribution of disconnected graph: that is, the probability of generating a graph should be within $(1 \pm \epsilon)$ of the correct probability.

Problem 3. Instead of counting cuts using the contraction algorithm, cycle coupling can be used as the starting point for analyzing graph sampling and network reliability. Write the *partition function* $z_G(p) = \sum p^{c_i}$ over all cut values c_i in G. Note that $u_G(p) \leq z_G(p)$.

(a) Argue that $z_G(p) = E[2^{N_G(p)-1} - 1]$, where $N_G(p)$ is a random variable denoting the number of connected components of $G(p)$. **Hint:** how many cuts does a k-vertex graph have?

(b) Use the cycle-coupling argument from class to conclude that over all n-vertex graphs with minimum cut c, the cycle (with $c/2$-edge "bundles" between neighboring vertices) maximizes $z_G(p)$ for any p.

(c) Show that for the cycle Y, $E[2^{N_Y(p)}] = (1 - p^{c/2})^n + (1 + p^{c/2})^n$ (**Hint:** write out the expectation in terms of the number of failed bundles and use the binomial theorem).

(d) Conclude that if $n^2 p^c \leq 1$ then $z_G(p) = O(n^2 p^c)$. In particular, $z_G(n^{-2}) = O(1)$.

(e) Conclude that there are at most $O(n^{2\alpha})$ cuts of value less than αc, proving cut counting from cycle coupling. **Hint:** what do such cuts contribute to $z_G(p)$?

(f) We analyzed graph-edge sampling using a union bound to a Chernoff bound on each cut. Writing out this sum of Chernoff bounds, argue that it has the form $z_G(q)$ for some $q < n^{-3/c}$, and is thus bounded by $O(1/n)$. This gives an alternate proof that the union bound is small.

Problem 4. Read through the handout on the final project, which is linked from the course website. Submit a paragraph-long (or so) proposal for the final project detailing the topic and scope of the proposed project. Include citations to relevant papers. If you are collaborating with other students, submit just one copy with all of your names on it. Working alone is permitted only with permission and with a good justification. Conversely, you should limit your groups to 3 students. Email your proposal to Professor Karger at `karger@mit.edu` and the TAs. Put "6.856 project" (without quotes) and the names of the group members in the subject line of your email (you won't get credit for a different subject line).

This proposal can be submitted late without penalty, but we can only give feedback after we have received it.

Problem 5—Optional. We explore a simpler reliability algorithm. Let $u_G(p)$ denote the probability that G becomes disconnected when edges fail with probability p. We wish to estimate this quantity. We know from class that Naive Monte Carlo provides an FPRAS when $p^c > n^{-3}$. Consider the following *estimator* for smaller p:

1. set $q \geq p$ such that $q^c = n^{-3}$
2. construct a graph H by contracting each edge of G with probability $1 - q$

3. compute and return $u_H(p/q)$

We will analyze this estimator.

(a) Prove that $E[u_H(p/q)] = u_G(p)$, where the expectation is over the random graph H we produce. That is, it is an unbiased estimator.

(b) Prove that H has $O(1)$ vertices w.h.p. (**Hint:** recall our coupling argument). Conclude that $u_H(p/q)$ can be computed quickly (by brute force) with high probability.

(c) Use the previous part to prove that $E[u_H(p/q)^2] = O((p/q)^{2c})$. **Hint:** condition on H having k vertices, what is the maximum possible $u_H(p/q)$ then?

(d) Conclude that the relative variance of the estimator is polynomial, and explain how that yields an FPRAS for $u_G(p)$.

The above analysis shows the algorithm runs in polynomial time w.h.p., but does not bound the expected runtime (perhaps it is exponential with some small probability). With more careful setting of parameters and analysis, you can improve the running time of this approach to $\tilde{O}(mn^2)$ in expectation and, with additional techniques, $\tilde{O}(n^2)$.