Counting and Sampling **Fall 2017**

Lecture 2: Monte Carlo Approach: An FPRAS for Network Unreliability

Lecturer: Shayan Oveis Gharan *September 29th*

Disclaimer: *These notes have not been subjected to the usual scrutiny reserved for formal publications.*

In this lecture we will discuss an algorithm for network unreliability problem. Given a network of n vertices where each link e disappears independently with probability p_e determine the probability that the surviving network is disconnected.

In this lecture for simplicity we assume that all p_e's are equal to p and we define $Fail(p)$ to denote the failure probability of the whole network. We prove the following theorem of Karger [Kar95]. Also, see [Kar16] for a much faster algorithm.

Theorem 2.1 ([Kar95]). *There is an FPRAS for the network reliability problem. That is given G, p and ϵ the algorithm returns a $1+\epsilon$ multiplicative approximation to $Fail(p)$ with probability $3/4$.*

Note that the success probability can be easily boosted up to $1-\delta$ by running $O(\log(1/\delta))$ independent copies of the algorithm and returning the median of the estimates.

Before discussing the above algorithm we discuss an algorithm of Karp, Luby and Madras for the DNF counting problem. Later we will see how we can employ DNF counting in network unreliability problem.

2.1 FPRAS for DNF Counting

A DNF is conjunction of clauses where each of them is a disjunction of literals, e.g., $(x_1 \wedge \overline{x_2} \wedge x_4) \vee (\overline{x_3} \wedge x_2) \dots$. Consider a DNF with n variables. Obviously the problem of finding a satisfying assignment for a DNF is in P. Here, we want to count the number of satisfying assignment to a DNF. Here we prove the following theorem:

Theorem 2.2 ([KLM89]). *There is an FPRAs for the DNF counting problem.*

Let's start by sampling. Suppose we choose N samples, $X^1, \dots, X^N$ where each $x^i_j \in \{0,1\}$ uniformly and independently at random. We output the fraction of X^i's that satisfy the formula times 2^n as our estimate. Let Y_i be the indicator that X^i is satisfying assignment and $Y = \frac{1}{N}(Y_1 + \dots + Y_N)$. Then, obviously,

$$\mathbb{E}[Y] = \frac{1}{N}\mathbb{E}[Y_1 + \dots + Y_N] = \frac{|S|}{2^n}$$

where S the set of all satisfying assignments. Now, we would like to argue that Y is close to $|S|/2^n$ with a high probability. Using Chernoff bound

$$\mathbb{P}[|Y - \mathbb{E}[Y]| > \alpha \mathbb{E}[Y]] \leq e^{-N\alpha^2 \mathbb{E}[Y]/3}.$$

If we want a $1 \pm \epsilon$ approximation of $|S|$ we need to we need $N > \frac{1}{\epsilon^2 \mathbb{E}[Y]}$ samples. So, if $\mathbb{E}[Y]$ is polynomially large we are done. But, $\mathbb{E}[Y]$ can be exponentially small in n. Note that the Hoeffding bound is essentially tight. Roughly speaking, if $\mathbb{E}[Y] \approx 2^{-\sqrt{n}}$ then most likely all $Y_i = 0$ unless N is exponentially large. In that case $Y = 0$ with high probability.

Now, we explain an algorithm of Karp, Luby and Madras [KLM89]. The idea is to decrease the size of the universe such that the target ratio is only inverse polynomially small. In fact, they consider a more general problem. Suppose we have sets $S_1, S_2, \ldots, S_m$ of a ground set of elements and we want to estimate $|\cup_i S_i|$. First, of all observe that we always have

$$\frac{1}{m} \leq \frac{|\cup_i S_i|}{\sum_i |S_i|} \leq 1$$

So, the idea is to construct an artificial universe U of size $|U| = \sum_i |S_i|$. How can we do that? It is enough that for each ground element e put a distinct copy of e inside each set S_i that contains e, so say if $e \in S_i$, we put (e, i) in S_i. Now, we can sample from U if we know $|S_i|$'s and we can sample efficiently from each of them:

First we sample i with probability proportional to $|S_i|$. Then, we sample a uniformly random element from S_i.

Now, having access to an element of U we just need to mark $|\cup_i S_i|$ elements of U as special and then use the Monte Carlo approach to estimate the ratio $|\cup_i S_i|/|U|$. For any element e we mark the smallest i such that $(e, i) \in U$. This way exactly $|\cup_i S_i|$ elements of U are marked and the Monte Carlo approach succeeds. Using only $O(\frac{m^2}{\epsilon^2} \log(1/\delta))$ samples from U we can estimate $\frac{|\cup_i S_i|}{|U|}$ within $1+\epsilon$ factor with probability $1-\delta$.

In our DNF counting problem each S_i corresponds to the set of assignments that satisfy the i-th clause. Obviously if the i-th clause has k literals, $|S_i| = 2^{n-k}$, and we can sample efficiently from S_i simply by fixing every literal that appears in S_i and choosing rest uniformly at random.

2.2 Back to Network Unreliability

First, let us discuss how the two problems are related. Suppose our graph have r cuts, $C_1, \ldots, C_r$. Then, we can use the above algorithm to estimate the probability that at all of the edges in one of these cuts fail. In particular, we write

$$\vee_{1 \leq i \leq r} \wedge_{e \in C_i} x_e$$

Say x_e is true if edge e fails. Then, every realization of failures that makes G disconnected correspond to a satisfying assignment for the above DNF formula. So, we can use our FPRAS to get a $1+\epsilon$ approximation to the probability that at least one of the cuts $C_1, \ldots, C_r$ fail.

Remark 2.3. *Note that there is a technical problem here, because each edge fails with probability p whereas in the proof of Theorem 2.2 we assumed that every variable is true/false with 1/2 probability. So, we slightly need to perturb the algorithm of Theorem 2.2 and weight each element according to the number of true variables. We leave this as an exercise.*

By the above discussion if G has polynomially many cuts then we are already done. The problem is that G has exponentially many cuts. Karger's idea is as follows: Let C denote the size of the minimum cut of G. Then, obviously,

$$Fail(p) \geq p^C =: q.$$

Now, we consider two cases

Case 1: $q > 1/\operatorname{poly}(n)$ In this case, we can simply use Monte Carlo simulation to obtain a $1+\epsilon$ approximation of $Fail(p)$. In particular, by Chernoff bound it is enough to generate $O(\log(1/\delta)/\epsilon^2 q)$ many samples to obtain a $1+\epsilon$ approximation to $Fail(p)$.

Case 2: $q < 1/poly(n)$ As it will be clear we need to let n^4 for poly(n). Here the idea is to divide all cuts of G into two groups: (i) Near minimum cuts and (ii) Large cuts. We use the following theorem of Karger to prove that there are only polynomialy many near minimum cut in any graph G. So, we can use Theorem 2.2 to estimate the probability that at least one of these cuts fails. To deal with large cuts we also use the following theorem to argue that it is very unlikely that any of the large cuts fails.

Theorem 2.4 (Karger [Kar95])**.** *For any graph G with n vertices and with minimum cut C, and for any $\alpha \geq 1$, the number of cuts of size at most αC in is at most $n^{2\alpha}$.*

The above theorem can be proved by Karger's contraction algorithm, see 521-Notes. Also, note that the statement of the above theorem is tight, as in a cycle of length n there are $O(n^{2\alpha})$ cuts with 2α edges.

Now, let α be a parameter that we fix later. We want to show that with probability at most ϵq all cuts of size at least αC survive.

Lemma 2.5. *Let $C_1, \ldots, C_r$ be all cuts of G and let us sort them in the order of their size*

$$|C_1| \leq |C_2| \leq \cdots \leq |C_r|.$$

For any $\alpha \geq 1$, and $q = n^{-\beta}$ we have

$$\mathbb{P}\left[\exists i \geq n^{2\alpha} : C_i \text{ fails}\right] \leq \frac{n^{2\alpha(-\beta/2+1)}}{\beta/2-1}$$

Proof. Firstly, by Theorem 2.4, for any $i = n^{2x}$, we have $|C_i| \geq xC$. In other words,

$$|C_i| \geq \frac{\log i}{2\log n} C$$

By union bound can write

$$\begin{aligned}
\mathbb{P}\left[\exists i \geq n^{2\alpha} : C_i \text{ fails}\right] &\leq \sum_{i \geq n^{2\alpha}} \mathbb{P}\left[C_i \text{ fails}\right] \\
&= \sum_{i \geq n^{2\alpha}} p^{|C_i|} \\
&\leq \sum_{i \geq n^{2\alpha}} p^{\frac{\log i}{2\log n} C} \\
&= \sum_{i \geq n^{2\alpha}} q^{\log(i)/2\log(n)}.
\end{aligned}$$

Say $q = n^{-\beta}$. We can write $q^{\log(i)/2\log(n)} = e^{-\beta log(n)\log(i)/2\log(n)} = i^{-\beta/2}$. Therefore,

$$\mathbb{P}\left[\exists i \geq n^{2\alpha} : C_i \text{ fails}\right] \leq \int_{x \geq n^{2\alpha}} x^{-\beta/2} dx = \frac{x^{-\beta/2+1}}{-\beta/2+1}\Big|_{n^{2\alpha}}^{\infty} = \frac{n^{2\alpha(\beta/2-1)}}{-\beta/2+1}$$

□

So, to make sure that all large cuts survive it is enough to choose α such that the above probability is at most ϵq. Observe that if we choose $\alpha = 2$ and making sure that $q \leq \epsilon \cdot n^{-4}$ we get

$$\mathbb{P}\left[\exists i \geq n^{2\alpha} : C_i \text{ fails}\right] \leq n^{-2\beta+4} \leq \epsilon q.$$

So, it is enough to solve the DNF counting problem for $n^{2\alpha} = n^4$ smallest cuts. These cuts can be found in polynomial time using Karger's contraction algorithm.

References

[Kar95] David R. Karger. A randomized fully polynomial time approximation scheme for the all terminal network reliability problem. In *STOC*, pages 11–17. ACM, 1995. 2-1, 2-3

[Kar16] David R. Karger. A fast and simple unbiased estimator for network (un)reliability. In *FOCS*, pages 635–644, 2016. 2-1

[KLM89] R. M. Karp, M. Luby, and N. Madras. Monte-carlo approximation algorithms for enumeration problems. *J. Algorithms*, 10(3):429–448, September 1989. 2-1, 2-2