CS 761: Randomized Algorithms Fall 2019

Lecture 8 — November 1, 2019

Prof. Gautam Kamath *By: Seung Gyu Hyun, Reza Karegar*
Edited by Vedat Levi Alev

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

Recall from last lecture:

Definition 1 (Hitting Time). h_{uv}*: The expected number of steps for a random walk starting at* u *to reach* v *for the first time.*

Definition 2 (Commute Time). c_{uv}*: The expected time for the random walk to reach* v *from* u *and then visit* u *again. That means:* $c_{uv} = h_{uv} + h_{vu}$

Definition 3 (Cover Time). $C(G)$*: Expected first time that all vertices are visited.*

Definition 4 (Effective Resistance). R_{uv}*: Resistance between vertices* u *and* v*.*

Theorem 5. $\forall u, v, c_{uv} = 2mR_{uv}$

Corollary 6. *For edge* $(u, v) = e \in E$*,* $c_{uv} \leq 2m$*.*

Proof. Since there exists an edge between u and v, $R_{uv} \leq 1$ and therefore $c_{uv} \leq 2m$. □

Theorem 7. *For the cover time* $C(G)$ *of a graph we have:* $C(G) \leq 2m(n-1) \leq n^3$

Proof. Construct T a spanning tree of G. To cover every vertex in G, we can consider the commute time for every adjacent pair of vertices in T. Therefore:

$$C(G) \leq \sum_{u,v \in T} C_{uv} \leq \sum 2m = 2m(n-1)$$

□

This bound is not tight in general. If we have a graph consisting of a clique of $n/2$ vertices and a line of the rest, then the cover time is n^3. Matthews showed that if we have a clique of n vertices, the cover time is $O(n \log n)$ [1].

Theorem 8. *Let* $R(G) = \max_{u,v} R_{u,v}$ *and* $C(G)$ *be cover time. We have:*

$$mR(G) \leq C(G) \leq 2e^3 mR(G) \ln n + n$$

Proof. We first prove the lower bound for $C(G)$, using the fact that $C(G)$ is at least as large as the hitting time for the hardest pair of vertices:

$$\begin{aligned} C(G) &\geq max\{h_{uv}, h_{vu}\} \\ &\geq \frac{C_{uv}}{2} \\ &= mR_{uv} \end{aligned}$$

The other direction: Fix some vertex s. $h_{*s} \leq 2mR(G)$, $\forall *$. With probability $\leq \frac{1}{e}$, a random walk has not been reached v after $2e^3 mR(G)$ (Markov's).

By Chernoff bound, haven't reached s after $2e^3 mR(G)\ln n$ steps with probability $\leq \frac{1}{n^3}$. Using union bound, we get that every vertex is visited with probability $\geq 1 - \frac{1}{n^2}$. Therefore, by using the previous n^3 upper bound for the case that happens with probability $1/n^2$, we have:

$$
\begin{aligned}
C(G) &\leq \left(1 - \frac{1}{n^2}\right) 2e^3 mR(G)\ln n + \left(\frac{1}{n^2}\right) n^3 \\
&\leq 2e^3 mR(G)\ln n + n
\end{aligned}
$$

□

1 Graph Connectivity

We are concerned with the problem of determining the connectivity of graph in logarithmic space: Given vertices $s, t \in V$ we would like to decide whether there is a path from s to t using logarithmic space only. We can use the following randomized procedure:

1. We start at s.
2. For $2n^3$ steps:
   (a) We perform a random walk and move to another vertex.
3. If t is seen, we output "yes", otherwise "no".

Using Markov's inequality, we can show that this algorithm succeeds with probability $\geq 1/2$. There is also a deterministic algorithm by Reingold that can achieve this deterministically [2].

1.1 Mixing time and coupling

In this document, we will use $\|\cdot\|$ for the *total variation distance* between two probability distributions. Up to a factor of 2, this is equivalent to the ℓ_1-distance between the vectors.

Definition 9 (Total variation distance)**.** $\|\boldsymbol{p} - \boldsymbol{q}\| = \frac{1}{2}\sum_{i\in S} |p_i - q_i|$, *where* $\boldsymbol{p}$ *and* $\boldsymbol{q}$ *are probability vectors.*

$\boldsymbol{\pi}$ is the stationary distribution of P ($\boldsymbol{\pi} = \boldsymbol{\pi}P$). Let $\boldsymbol{p}_X^{(t)}$ be the distribution after t steps starting at X, $\Delta_X(t) = \|\boldsymbol{p}_X^{(t)} - \boldsymbol{\pi}\|$ and $\Delta(t) = \max_{X\in S} \Delta_X(t)$. Furthermore, let $Y_X(\epsilon) = \min\{t : \Delta_X(t) \leq \epsilon\}$ and $Y(\epsilon) = \max_{X\in S} Y_X(\epsilon)$. The Markov chain described by P is rapidly mixing if $Y(\epsilon)$ is polynomial in $\ln(1/\epsilon)$ and the number of states.

Assume M_t is a Markov chain with state space S. Let Z_t be a jointly distributed pair of random variables, each of which obey the law of the chain when observed individually. In particular, X_t, Y_t and M_t all follow the same law. $Z_t = (X_t, Y_t)$ is a coupling on $S \times S$ iff:

1. $\Pr[X_{t+1} = x' | Z_t = (x, y)] = \Pr[M_{t+1} = x' | M_t = x]$
2. $\Pr[Y_{t+1} = y' | Z_t = (x, y)] = \Pr[M_{t+1} = y' | M_t = y]$

Example 10. *This condition always holds for $(x, x) \to (x', x')$, i.e., when coupling a Markov chain with itself.*

1.2 Coupling Lemma

Assume $Z_t = (X_t, Y_t)$ is a coupling of Markov chains X and Y. Then:

$$Y(\epsilon) \leq \min\{T \geq 0 : \Pr[X_T \neq Y_T \mid X_0 = x, Y_0 = y] \leq \epsilon, \forall x, y \in S\}$$

Intuition: The probability of X_T and Y_T disagreeing on step T can be used to bound how far the Markov chain is from mixing on step T.

1.3 Random walk on hypercube

Consider a d-dimensional hypercube whose each corner is in $\{-1, +1\}^d$.

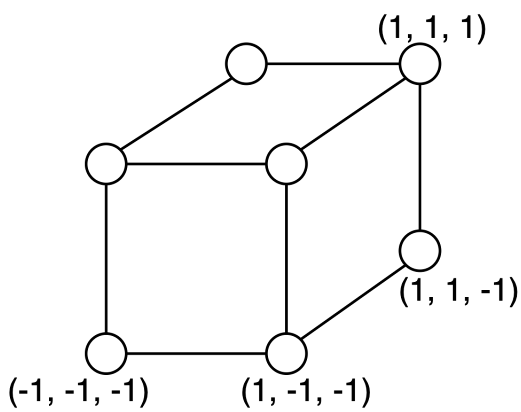

Figure 1: A 3-dimensional hypercube

We can define the following lazy walk on the hypercube, which gives us the Markov chain M_t:

At step $(t + 1)$:

1. We pick a random coordinate j.
2. We set the value of the j-th coordinate in M_t to 1 with probability 1/2 and to -1 with probability 1/2, and this gives us M_{t+1}.

Remark 11. *This Markov Chain is irreducible, aperiodic and uniform, therefore stationary.*

To analyze the mixing time, we define the following coupling:

1. X_t: We pick coordinate i uniformly at random and set it randomly to v.
2. Y_t: We pick coordinate i and set it to v (the same coordinate and value as X_t).

Example 12 (One possible step of this coupling)**.**

$$X_0 = (-1, -1, -1), Y_0 = (1, 1, 1), i = 3, v = -1$$

$$X_1 = (-1, -1, -1), Y_1 = (1, 1, -1)$$

Observation 13. *A coordinate that has become the same, will always remain the same.*

Observation 14. *The time taken to select every coordinate at least once is a coupon collector problem with d coupons, which means that it takes on average $d \log d$ time to select all coordinates. Therefore:*

$$Y(\epsilon) \leq O(d \log d + d \log(1/\epsilon))$$

1.4 Shuffling n cards

We describe the following Markov chain to shuffle n cards:

1. We pick a random card (value $= c$).
2. We move the selected card to the top of deck.

Coupling: The coupling is done similar to the previous example, with the difference that now X_t and Y_t coordinate on the value of c instead of i and v, i.e., they both move the same card to the top of deck at each step.

Observation 15. *After picking c and moving it to the top of deck, it remains level in both chains. We could again use the coupon collector argument, since we're looking for the expected time required to select every card at least once. Therefore:*

$$Y(\epsilon) \leq O(n \log(n/\epsilon))$$

1.5 Independent sets of fixed size

Definition 16 (Independent Set)**.** *An independent set of size k is a subset of k vertices in a graph, none of which are adjacent.*

Example 17 (Figure 2)**.** *Independent Set of size* 3

We describe a Markov chain to sample an independent set of size k. X_t denotes the vertices in the independent set at step t. To reach X_{t+1}, we do as follows:

1. We choose $v \in X_t$ and $w \in V$ uniformly at random.
2. If $w \notin X_t$ and $X_t - v + w$ is an independent set, $X_{t+1} = X_t - v + w$, otherwise $X_{t+1} = X_t$.

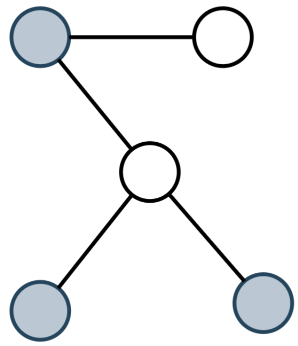

Figure 2: Independent set of size 3

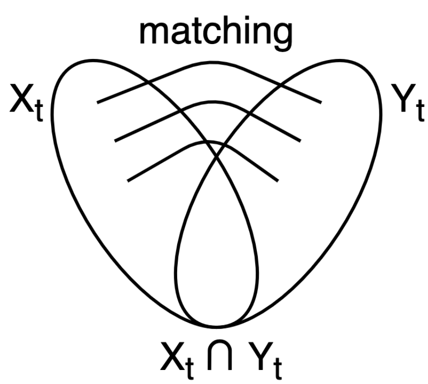

Figure 3: Independent sets and a perfect matching M between them

We let n denote the number of vertices and Δ the maximum degree of a vertex. If $k \leq \frac{n}{3\Delta+3}$, then this Markov chain is rapidly mixing.

Coupling:

1. For chain X, we choose $v \in X_t$ and $w \in V$ at random, then make a move.
2. For chain Y, we look at the vertex v selected for X:
   (a) If $v \in X_t \cap Y_t$: We use the same v and w and try to make a move. (May not be able to make the same move in both X and Y)
   (b) If $v \notin X_t \cap Y_t$: We use $M(v)$ and w and try to make a move. ($M(v)$ denotes the perfect matching between the vertices in $X_t - (X_t \cap Y_t)$ and $Y_t - (X_t \cap Y_t)$)

Analysis: Let $d_t = |X_t - Y_t|$ denote the Hamming distance between the two sets at step t. We want d_t to be equal to 0, which indicates that the to Markov chains reaching the same state.

Observation 18. $d_{t+1} \in \{d_t - 1, d_t, d_t + 1\}$

Question 1. *How can we have $d_{t+1} = d_t + 1$? This can only happen if $v \in X_t \cap Y_t$, and w is adjacent to $X_t - Y_t$, but not $Y_t - X_t$ (or vice versa).*

In these cases w is a vertex or neighbor of $(X_t - Y_t) \cup (Y_t - X_t)$, therefore:

$$\Pr[d_{t+1} = d_t + 1 | d_t > 0] \leq \left(\frac{k - d_t}{k}\right)\left(\frac{2d_t(\Delta + 1)}{n}\right)$$

Here, the first term relates to picking v in the intersection, and the second term to selecting w.

Question 2. *How can we have $d_{t+1} = d_t - 1$? This can only happen if $v \notin Y_t$ and w is neither adjacent to or equal to a vertex in $X_t \cup Y_t - \{v, M(v)\}$.*

Therefore:

$$\Pr[d_{t+1} = d_t - 1 | d_t > 0] \geq \left(\frac{d_t}{k}\right)\left(\frac{n - (k + d_t - 2)(\Delta + 1)}{n}\right)$$

Thus:

$$\begin{aligned}
E[d_{t+1} | d_t] &= \Pr[d_{t+1} = d_t + 1](d_t + 1) + \Pr[d_{t+1} = d_t - 1](d_t - 1) + \Pr[d_{t+1} = d_t]d_t \\
&\leq d_t + \left(\frac{k - d_t}{k}\right)\left(\frac{2d_t(\Delta + 1)}{n}\right) - \left(\frac{d_t}{k}\right)\left(\frac{n - (k + d_t - 2)(\Delta + 1)}{n}\right) \\
&= d_t\left(1 - \frac{n - (3k - d_t - 2)(\Delta + 1)}{kn}\right) \\
&\leq d_t\left(1 - \frac{n - (3k - 3)(\Delta + 1)}{kn}\right)
\end{aligned}$$

Now we use induction on t:

$$E[d_t] \leq d_0\left(1 - \frac{n - (3k - 3)(\Delta + 1)}{kn}\right)^k, \; d_0 < k,$$

$$\begin{aligned}
\Pr[d_t \geq 1] &\leq E[d_t] \\
&\leq k\left(1 - \frac{n - (3k - 3)(\Delta + 1)}{kn}\right)^t \\
&\leq k \exp\left(\frac{-t(n - (3k - 3)(\Delta + 1))}{kn}\right) \Rightarrow \\
\Pr[d_t \geq 1] &\to 0
\end{aligned}$$

Therefore:

$$Y(\epsilon) \leq \frac{kn \ln(k/\epsilon)}{n - (3k - 3)(\Delta + 1)}$$

2 Monte Carlo

2.1 Estimating value of π

Assume we want to estimate the value of π using Monte Carlo techniques. We start by drawing uniformly random samples from the square in Figure 4 and count the number of samples inside the circle.

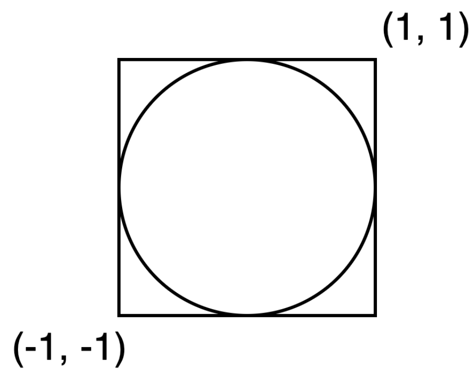

Figure 4

We note that the area of the square and circle are 4 and $\pi r^2 = \pi$ respectively. Therefore, the probability of one of the points being inside the circle is $\frac{\pi}{4}$ and E[Fraction of points in the circle] $= \frac{\pi}{4}$. If w samples out of the m samples fall inside the circle, $4\frac{m}{w}$ could be used as an estimate for the value of π.

Theorem 19. *Let $X_1, \dots, X_m$ be i.i.d. Bernoulli random variables, $E[X_i] = \mu$. If $m \geq \frac{3\ln(2/\delta)}{\epsilon^2 \mu}$:*

$$\Pr\left[\left|\frac{1}{m}\sum X_i - \mu\right| \geq \epsilon\mu\right] \leq \delta$$

Definition 20. *A random algorithm gives an (ϵ, δ)-approximation for V if output X is:*

$$\Pr[|X - V| \leq \epsilon V] \geq 1 - \delta$$

Definition 21. *A fully polynomial randomized approximate scheme (FPRAS) is a random algorithm if given x as input and $0 < \epsilon < 1$, outputs an $(\epsilon, \frac{1}{4})$-approx to $V(x)$ in time poly($\frac{1}{\epsilon}$, size of x).*

2.2 DNF Counting

We start by recalling that CNF (Conjunctive Normal Form) is a canonical normal form of a logical formula consisting of a conjunction of disjunctions. Conversely, DNF (Disjunctive Normal Form) is another canonical normal form which consists of a disjunction of conjunctions.

Example 22 (CNF)**.** $(X_1 \vee X_2 \vee X_3) \wedge (\overline{X_1} \vee \overline{X_2} \vee X_3)$

Example 23 (DNF)**.** $(Y_1 \wedge Y_2 \wedge Y_4) \vee (Y_3 \wedge \overline{Y_4} \wedge Y_1)$

Remark 24. *Counting the number of satisfying assignments in a DNF problem is #P hard.*

We can design a (ϵ, δ)-approx algorithm with running time $\Omega\left(\frac{\ln(1/\delta)}{\epsilon^2 \mu}\right)$ by sampling variable assignments uniformly at random. However, a question arises regarding this approach:

Question 3. *What if μ is small? For $O(n^2)$ satisfying assignments, $\mu = \frac{n^2}{2^n}$ which makes the number of required samples exponentially large.*

To overcome this problem, we assume $F = C_1 \vee C_2 \vee \cdots \vee C_m$. Let SC_i be set of satisfying assignments of clause i. We have $|SC_i| = 2^{n-l_i}$, where $l_i =$ number of literals in clause i. We want to approximate $|\bigcup SC_i|$.

Let $u = \{(i,a) | 1 \le i \le m, a \in SC_i\}$. Therefore: $|u| = \sum_{i=1}^{m} |SC_i|$.

$S = \{(i,a) | 1 \le i \le m, a \in SC_i,\ but\ a \notin SC_j\ for\ j < i\}$.

We sample from u and check if the element is in S. This gives an approximation for $\frac{|S|}{|u|}$

This is a good estimate (polynomial queries) if $\frac{|S|}{|u|}$ is large ($\frac{|S|}{|u|} \ge \frac{1}{m}$). $O(m \log(1/\delta)/\epsilon^2)$ samples suffice.

Question 4. *How to sample from u?*

1. We pick clause i with probability $|SC_i| / \sum_{j=1}^{m} |SC_j|$.
2. We pick random satisfying assignment in SC_i.

$$
\begin{aligned}
\Pr[(i,a)\ is\ chosen] &= \Pr[i\ is\ chosen] \cdot \Pr[a\ is\ chosen] \\
&= \frac{|SC_i|}{|u|} \cdot \frac{1}{|SC_i|} \\
&= \frac{1}{|u|}
\end{aligned}
$$

2.3 Percolation

Let G be a graph and suppose each edge in G is removed with probability p.

Question 5. *What is the probability that the graph is still connected?*

We define $FAIL(p)$ as the probability that the graph is disconnected after removing each edge with probability p.

Lemma 25. *The graph is disconnected, if and only if there exists a cut in the initial graph which all of its edges have failed.*

Define c as the min-cut size. We divide the proof to two cases:

1. If $p^c \ge n^{-4}$: In this case $FAIL(p) \ge n^{-4}$ and therefore we can use a $O\left(\frac{n^4 \log(1/\delta)}{\epsilon^2}\right)$-approx simple sampling algorithm.
2. If $p^c < n^{-4}$: We ignore the "big" cuts and enumerate all the "small" cuts.

   Claim 26. *An undirected graph has $O(n^{2\alpha})$ cuts with $\le \alpha c$ edges.*

We can enumerate all of these small cuts in $\tilde{O}(n^{2\alpha} \cdot T - cont)$ time, where $T - cont$ is the running time of the contraction algorithm.

To ignore the big cuts, we need to upper bound the probability of any one of them failing. Let $p^c = n^{-(2+k)}$ for some $k > 2$. We have:

$$\begin{aligned} \Pr[\text{Some cut with} \geq \alpha c \text{ edges fails}] &\leq \int_{\beta \geq \alpha} O(n^{2\beta}) p^{\beta c} \, d\beta \\ &\leq \int_{\beta \geq \alpha} n^{-k\beta} \, d\beta \\ &\leq O(n^{-k\alpha}) \end{aligned}$$

By setting $\alpha = 2t \frac{\ln(\epsilon / O(1))}{k \ln n}$ we get:

$$O(n^{-k\alpha}) \leq n^{-2k}\epsilon \leq \epsilon \cdot FAIL(p)$$

2.3.1 Reduction to DNF for small cuts

(a) We define variable x for every edge e.

(b) For each small cut, we make a clause. For example:

$$(x_{e_1} \wedge x_{e_2} \wedge \cdots \wedge x_{e_k}) \vee \ldots$$

Now, we set p as the probability of each variable being satisfied, which is the difference from the regular DNF counting problem.

References

[1] Peter Matthews. Covering problems for brownian motion on spheres. *Ann. Probab.*, 16(1):189–199, 01 1988. doi: 10.1214/aop/1176991894. URL `https://doi.org/10.1214/aop/1176991894`.

[2] Omer Reingold. Undirected connectivity in log-space. *J. ACM*, 55(4):17:1–17:24, September 2008. ISSN 0004-5411. doi: 10.1145/1391289.1391291. URL `http://doi.acm.org/10.1145/1391289.1391291`.