COMPSCI 690RA: Randomized Algorithms and Probabilistic Data Analysis

Prof. Cameron Musco

University of Massachusetts Amherst. Spring 2022.

Lecture 12 (Final Lecture!)

Logistics

- The final exam is this Friday 5/5 at 10:30am in this room.
- I will hold extended office hours today from 2-4pm and tomorrow from 4-6pm.
- I will accept final project submissions up until Sunday 5/8 at 11:59pm.
- Please complete your SRTI for the class when you get a chance!

Summary

Last Week: Finish up Markov Chains Unit.

- Mixing time analysis via coupling.
- Example applications to shuffling and random walks on the hypercube.
- Markov Chain Monte Carlo (MCMC) methods.
- Example of reductions from counting to sampling (e.g., for counting independent sets).

Today: The Probabilistic Method (not on the exam)

- From probabilistic proofs to algorithms via the method of conditional expectations.
- The Lovasz local lemma for events with 'bounded' correlation.
- Entropic proof of the algorithmic LLL.

First...a detour

The St. Petersburg Paradox

Consider the following game: you keep flipping a fair coin, until it hits tails. You win $\$2^{k+1}$, where k is the number of heads you see.

Let X be the amount of money you win. What is $\mathbb{E}[X]$?

How much money would you pay to play this game? Why?

One Solution to the Paradox: The expected value of the game is not $\mathbb{E}[X]$, but $\mathbb{E}[U(X)]$ where U is some utility function.

$U(\cdot)$ determines how much actual value you derive from a given amount of money. We expect generally that U is concave – diminishing marginal utility.

Maximizing Expected Log Winnings

What is $\mathbb{E}[U(\mathsf{X})] = \mathbb{E}[\log_2(\mathsf{X})]$ for our game?

A More 'Realistic' Scenario

You are given \$25 and are allowed to play the following game repeatedly: You have a biased coin that hits heads 60% of the time. You can wager \$$w$ on if the coin hits heads or tails. If you are correct, you win \$$w$, and if you are incorrect, you lose \$$w$.

How should you determine the size of your bets?

$$\mathbb{E}[\log(X_{i+1})|X_i] = .6 \cdot \log(X_i + w) + .4 \cdot \log(X_i - w).$$

Write $w = r \cdot X_i$. Then:

$$\begin{aligned}\mathbb{E}[\log(X_{i+1})|X_i] &= .6 \cdot \log(X_i \cdot (1 + r)) + .4 \cdot \log(X_i \cdot (1 - r)) \\ &= \log(X_i) + .6 \log(1 + r) + .4 \log(1 - r).\end{aligned}$$

To maximize $.6 \log(1 + r) + .4 \log(1 - r)$, set its derivative to 0:

$$0 = \frac{.6}{1 + r} - \frac{.4}{1 - r}.$$

Optimal $r = 0.2$. I.e., you should bet 20% of your money each time.

Kelly Criterion

The prior analysis is a special case of the Kelly criterion.

$$r = p - \frac{q}{b}.$$

Lots of interesting topics here, closely related to Markov chains and Martingales.

The Probabilistic Method

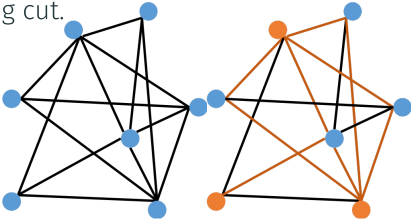
The Basic Idea: Suppose we want to prove that a combinatorial object satisfying a certain property exists. Then it suffices to exhibit a random process that produces such an object with probability > 0.

A common tool: For a random variable with $\mathbb{E}[X = \mu]$, $\Pr[X \geq \mu] > 0$ and $\Pr[X \leq \mu] > 0$.

Example 1: Max-Cut

Prove that for any graph with m edges, there exists a cut containing at least $m/2$ edges.

Consider a random partition of the nodes (each node is included independently in each half with probability 1/2). Let X be the size of the corresponding cut.



We have $\mathbb{E}[X] =$

Therefore, $\Pr[X \geq m/2] > 0$. So every graph with m edges has a cut containing at least $m/2$ edges.

Example 2: 3-SAT

Prove that for any 3-SAT formula, there is some assignment of the variables such that at least 7/8 of the clauses are true.

Consider a random assignment of the variables. And let X be the number of satisfied clauses.

$$(x_1 \vee \bar{x}_2 \vee x_4) \wedge (x_2 \vee \bar{x}_4 \vee x_3) \wedge (x_2 \vee x_3 \vee x_4) \wedge \ldots$$

What is $\mathbb{E}[X]$?

So, $\Pr[X \geq 7/8m] > 0$. So there is an assignment satisfying at least 7/8 of the clauses in every 3-SAT formula.

From Existence to Efficient Algorithms

Max-Cut Approximation: A randomly sampled partition cuts $m/2$ edges in expectation. But how many partitions do we need to sample before finding a cut of size at least $m/2$ with good probability?

Let p be the probability of finding a cut of size $\geq m/2$. Then:

$$\mathbb{E}[X] = \frac{m}{2} \leq (1-p) \cdot \left(\frac{m}{2} - 1\right) + p \cdot m$$
$$\implies \frac{1}{\frac{m}{2}+1} \leq p.$$

How many attempts do we need to take to find a large cut with probability at least $1-\delta$? $O(m \cdot \log(1/\delta))$

We can also derandomize this algorithm in a very simple way.

Let $x_1, x_2, \ldots \in \{0, 1\}$ indicate if the vertices are included on one side of the random partition.

Consider determining these randsom variables sequentially.

$$\frac{m}{2} = \mathbb{E}[X] = \frac{1}{2}\mathbb{E}[X|x_1 = 1] + \frac{1}{2}\mathbb{E}[X|x_1 = 0].$$

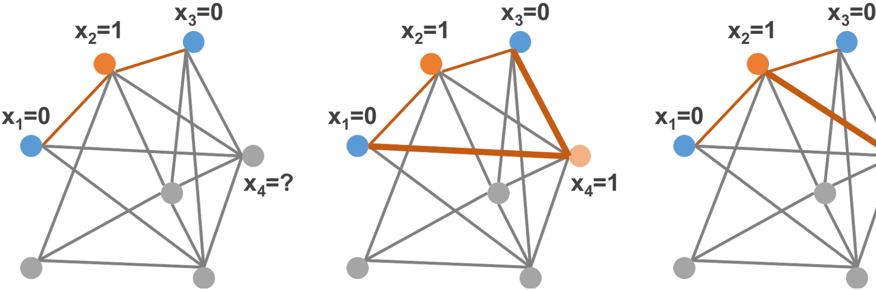
Set $x_1 = v_1$ such that $\mathbb{E}[X|x_1 = v_1] \geq \frac{m}{2}$ Then we have:

$$\frac{m}{2} \leq \mathbb{E}[X|x_1 = v_1] = \frac{1}{2}\mathbb{E}[X|x_1 = v_1, x_2 = 1] + \frac{1}{2}\mathbb{E}[X|x_1 = v_1, x_2 = 0]$$

Set $x_2 = v_2$ such that $\mathbb{E}[X|x_1 = v_1, x_2 = v_2] \geq \frac{m}{2}$. And so on...

Conditional Expectations for Cuts

How can we pick v_i such that $\mathbb{E}[X|x_1 = v_1, \ldots, x_{i-1} = v_{i-1}] \geq \frac{m}{2}$?



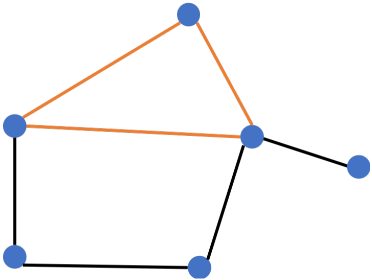
$\mathbb{E}[X|x_1 = 0, \ldots, x_4 = 1] = \frac{1}{2} \cdot 10 + 2 = 7 \mathbb{E}[X|x_1 = 0, \ldots, x_4 = 0] = \frac{1}{2} \cdot 10 + 1 = 6$

Natural greedy approach: add vertex i to the side of the cut to which it has fewest edges.

Yields a 1/2 approximation algorithm for max-cut. 16/17 is the best

Large Girth Graphs

The girth of a graph is the length of its shortest cycle.

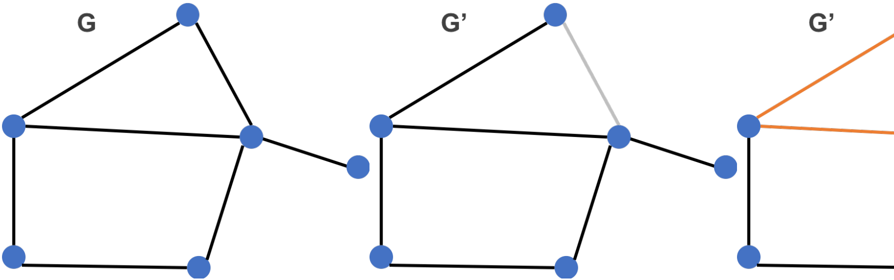


Natural Question: How large can the girth be for a graph with m edges?

Erdös Girth Conjecture: For any $k \geq 1$, there exists a graph with $m = \Omega(n^{1+1/k})$ edges and girth $2k+1$.

Relevance to Spanners

A spanner is a subgraph that approximately preserves shortest path distances. We say G' is a spanner for G with stretch t if for all u, v $d_{G'}(u, v) \leq t \cdot d_G(u, v)$.



Even when G' excludes a single edge, $t \geq girth(G) - 1$.

Erdös Girth Conjecture $\implies$ there are no generic spanner constructions with $o(n^{1+1/k})$ edges and stretch $\leq 2k - 1$.

Large Girth Graphs via Probabilistic Method

Theorem

For any fixed $k \geq 3$, there exists a graph with n nodes, $\Omega(n^{1+1/k})$ edges, and girth $k+1$.

Sample and Modify Approach: Let G be an Erdös-Renyi random graph, where each edge is included independently with probability $p = n^{1/k-1}$. Remove one edge from every cycle in G with length $\leq k$, to get a graph with girth $k+1$.

Let X be the number of edges in the graph and Y be the number of cycles of length $\leq k$. Suffices to show $\mathbb{E}[X - Y] = \Omega(n^{1+1/k})$.

$$\mathbb{E}[X] = \frac{n(n-1)}{2} \cdot p = \frac{1}{2} \cdot \left(1 - \frac{1}{n}\right) \cdot n^{1+1/k}.$$

$$\mathbb{E}[Y] = \sum_{i=3}^{k} \binom{n}{i} \cdot \frac{(i-1)!}{2} \cdot p^i \leq \sum_{i=3}^{k} n^i p^i = \sum_{i=3}^{k} n^{i/k} < k \cdot n.$$

Large Girth Graphs via Probabilistic Method

So far: An Erdös-Renyi random graph with $p = n^{1/k-1}$ has expected number of edges (X) and cycles of length $\leq k-1$ (Y) bounded by:

$$\mathbb{E}[X] = \frac{1}{2} \cdot \left(1 - \frac{1}{n}\right) \cdot n^{1+1/k}$$
$$\mathbb{E}[Y] < k \cdot n.$$

When k is fixed and n is sufficiently large, $k \cdot n \ll n^{1+1/k}$. Thus,

$$\mathbb{E}[X - Y] = \Omega(\mathbb{E}[X]) = \Omega(n^{1+1/k}),$$

proving the theorem.

Lovasz Local Lemma

Probabilities of Correlated Events

Suppose we want to sample a random object that avoids n 'bad events' $E_1, \ldots, E_n$.

E.g., we want to sample a random assignment for variables that satisfies a a k-SAT formula with n clauses. E_i is the event that clause i is not satisfied.

If the E_i are independent, and $\Pr[E_i] < 1$ for all i then:

$$\Pr\left[\neg \bigcup_{i=1}^{n} E_i\right] = \prod_{i=1}^{n} (1 - E_i) > 0.$$

What if the events are not independent?

If $\sum_{i=1}^{n} \Pr[E_i] < 1$ then by a union bound,

$$\Pr\left[\neg \bigcup_{i=1}^{n} E_i\right] \geq 1 - \sum_{i=1}^{n} > 0.$$

As n gets large, the union bound gets very weak – each event has to occur with probability $< 1/n$ on average.

Bounded Correlation

Consider events $E_1, \ldots, E_n$ where E_i is independent of any $j \notin \Gamma(i)$ (the neighborhood of i in the dependency graph

E.g., consider randomly assigning variables in a k-SAT formula with n clauses, and let E_i be the event that clause i is unsatisfied.

$$(x_1 \vee \bar{x}_2 \vee x_3) \wedge (x_2 \vee \bar{x}_4 \vee x_3) \wedge (x_4 \vee x_5 \vee x_6) \wedge (\neg x_4 \vee x_6 \vee x_7) \ldots$$

Theorem (Lovasz Local Lemma)

Suppose for a set of events $E_1, E_2, \ldots, E_n$, $\Pr[E_i] \leq p$ for all i, and that each E_i is dependent on at most d other events E_j (i.e., $|\Gamma(i)| \leq d$, then if $4dp \leq 1$:

$$\Pr\left[\neg \bigcup_{i=1}^{n} E_i\right] > (1-2p)^n > 0.$$

In the worse case, $d = n - 1$ and this is similar to the union bound. But it can be much stronger.

LLL Application: k-SAT

Theorem

If no variable in a k-SAT formula appears in more than $\frac{2^k}{4k}$ clauses, then the formula is satisfiable.

Let E_i be the event that clause i is unsatisfied by a random assignment. $\Pr[E_i] \leq \frac{1}{2^k} = p$.

$|\Gamma(i)| \leq k \cdot \frac{2^k}{4k} = \frac{2^k}{4} = d$

So $4dp = 4 \cdot \frac{1}{2^k} \cdot \frac{2^k}{4} \leq 1$, and thus $\Pr\left[\neg \bigcup_{i=1}^{n} E_i\right] > 0$. I.e., a random assignment satisfies the formula with non-zero probability.

Algorithmic LLL

Important Question: Given an Lovasz Local Lemma based proof of the existence, can we convert it into an efficient algorithm?

Moser and Tardos [2010] prove that a very natural algorithm can be used to do this.

Let $E_1, \ldots, E_n$ be events determined by a set of independent random variables $V = \{v_1, \ldots, v_m\}$. Let $v(E_i)$ be the set of variables that E_i depends on.

Resampling Algorithm:

1. Assign $v_1, \ldots, v_m$ random values.
2. While there is some E_i that occurs, reassign random values to all varables in $v(E_i)$.
3. Halt when an assignment is found such that no E_i occurs.

Algorithmic LLL

Theorem (Algorithmic Lovasz Local Lemma)

Consider a set of events $E_1, E_2, \ldots, E_n$ determined by a finite set of random variables V. If for all i, $\Pr[E_i] \leq p$ and $|\Gamma(i)| \leq d$, and if $ep(d+1) \leq 1$, then RESAMPLING *finds an assignment of the variables in V such that no event E_i occurs. Further, the algorithm makes $O(\frac{n}{d})$ iterations in expectation.*

Application to k-SAT: Consider a k-SAT formula where no variable appears in more than $\frac{2^k}{5k}$ clauses. Let E_i be the event that clause i is unsatisfied by a random assignment

$$\Pr[E_i] \leq \frac{1}{2^k} = p \quad \text{and} \quad |\Gamma(i) \leq k \cdot \frac{2^k}{5k} = \frac{2^k}{5} = d.$$

Have $ep(d+1) \leq \frac{e}{5} + \frac{e}{2^k} \leq 1$ as long as $k \geq 3$, so the theorem applies, giving a polynomial time algorithm for this variant of k-SAT.

Entropic Proof of Algorithmic LLL

Moser's 'entropic proof' of the algorithmic LLL uses a particularly cool technique.

Focus on the case of k-SAT where $|\Gamma(i)| < d = \frac{2^k}{8} = 2^{k-3}$.

- In each iteration of rerandomization, the algorithm uses k random bits. So for T iterations it uses Tk random bits.
- We will show that if we run the algorithm for too long, then we obtain a compression of these bits into a string of $< Tk$ bits, which shouldn't be possible (since they are random bits and incompressible).

Incompressibility Fact: For any function f mapping inputs in $\{0,1\}^t$ to distinct, possibly variable length binary output strings, if s is a uniform random t-bit binary strong, then for any integer c, $\Pr[length[f(s)] \leq t - c] \leq \frac{1}{2^{c-1}}$.

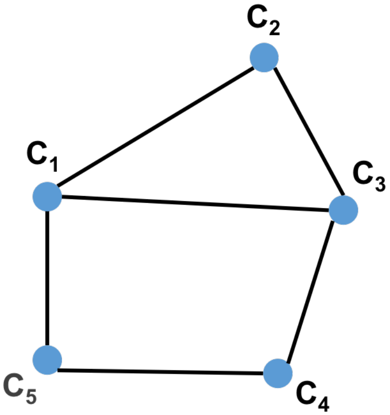
Compressing Bits While Solving k-SAT

- Initialize random assignments for the m variables using m bits.
- Iterate through the clauses, recording '1' for each that is satisfied, and recording '0' when you reach an unsatisfied clause i.
- Run LOCALCORRECT(i). Then move on to the next clause.
- After completion of all clauses, record the final state of the m variables using m bits.

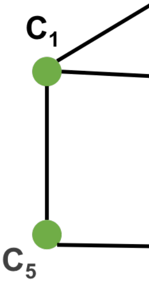
LOCALCORRECT(i):

- Resample random values for the variables in clause i, using k bits (but don't record them!).
- While some clause $j \in \Gamma(i) \cup \{i\}$ is unsatisfied, pick the first such j, and record '0' along with j using $k - 3$ bits. Then run LOCALCORRECT(j).
- Record '1' upon termination.

Compression Illustration



Record:
Bits Used:



Record:
Bits Used:

Compressing Bits While Solving k-SAT

Claim 1: If the algorithm runs for T iterations, it uses $m + Tk$ random bits.

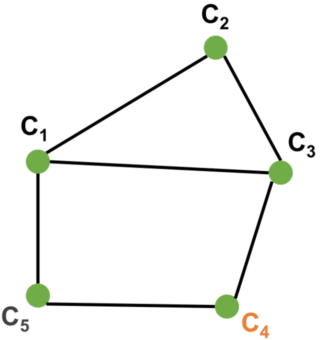
Claim 2: If the algorithms runs for T iterations it records a transcript of $m + n + T(k-1)$ bits.

Claim 3: The random bits used by the algorithm can be recovered uniquely from the transcript.

We can work backward from the final state, recovering the state of the variables at each step, and hence all the random bits. Critically, when a clause is rerandomized, we know exactly how its bits were been set before rerandomization (there is just a single unsatisfying assignment for the clause).

So we have compressed $B = m + Tk$ bits to $B' = m + n + T(k-1)$ bits. Setting, $T = n + c$ we have $B = m + Tk$ and $B' = m + Tk - c$. Thus, by our incompressibility claim, the algorithm must terminate in $n + c$ steps with probability $1 - 1/2^{c-1}$.

Recovery Illustration



Record: 10110000001…10111011
Bits Used: 10110001,10111,00100,10101,10111

Thanks for a great semester!