9
MCMC, Sampling, and Counting

9.1 Introduction

This lecture explores the deep connection between sampling from a distribution and approximately counting the size of a set. We will cover:

- Markov Chain Monte Carlo (MCMC).
- The connection between Sampling and Counting.
- Unbiased Estimators and Approximation Schemes (FPRAS).
- Application: Counting Colorings via Samplers.

9.2 The Monte Carlo Method and Unbiased Estimators

Consider the following problem:

Problem 9.1 (Counting DNF Assignments)**.** Given a Boolean formula φ in Disjunctive Normal Form (DNF), how many satisfying assignments does it have?

A DNF formula is an OR of ANDs (clauses), e.g., $\varphi = (x_1 \wedge x_3 \wedge \bar{x}_7) \vee (x_2 \wedge \bar{x}_9 \wedge x_{13}) \vee \dots$.

Note the contrast with Conjunctive Normal Form (CNF). For CNF, counting the number of satisfying assignments is at least as hard as determining if the count is > 0, which is the SAT problem (NP-complete). However, determining satisfiability for DNF is easy: φ is satisfiable if any single clause is satisfiable. Counting, however, remains difficult (#P-complete).

9.2.1 A Simple Monte Carlo Algorithm

We can estimate the number of satisfying assignments using the **Monte Carlo** paradigm. Let S be the set of satisfying assignments, and $U = \{T, F\}^n$ be the universe of all assignments ($|U| = 2^n$).

Algorithm 4: Simple Monte Carlo for DNF Counting

```
Initialize count C = 0
for i = 1 to N do
    Pick an assignment x ∈ U uniformly at random
    if x satisfies φ (i.e., x ∈ S) then
        C ← C + 1
Output Estimate X̂ = 2^n · C/N
```

Example 9.2 (Another simple example of Monte Carlo Method: Estimating π)**.** We can estimate π using Monte Carlo. Consider the square $[-1,1]^2$. Pick a random point $(x,y) \in [-1,1]^2$. Check if

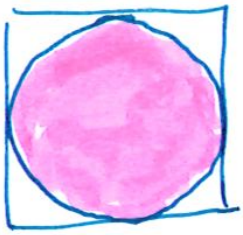

Figure 9.1: Estimating π via Monte Carlo.

$x^2 + y^2 \leq 1$ (the point is in the unit circle). The probability that the point is in the circle is $p = \frac{\text{Area}(Circle)}{\text{Area}(Square)} = \frac{\pi \cdot 1^2}{2^2} = \frac{\pi}{4}$. By repeating this N times and taking the fraction of successes, we can estimate p, and thus estimate π.

9.2.2 *Analysis and FPRAS*

How many samples N do we need? We typically want a multiplicative approximation with high probability.

Definition 9.3 (FPRAS)**.** A Fully Polynomial Randomized Approximation Scheme (FPRAS) for estimating a value V is an algorithm that, given inputs $\epsilon > 0$ and $\delta > 0$, outputs an estimate X such that

$$\Pr(|X - V| \leq \epsilon V) \geq 1 - \delta.$$

The runtime must be polynomial in the instance size, $1/\epsilon$, and $\log(1/\delta)$.

Let's analyze the simple Monte Carlo algorithm. We are estimating $p = \frac{|S|}{|U|}$. Let X_i be an indicator variable for the success of the i-th trial. $E[X_i] = p$. Let $Z = \sum_{i=1}^{N} X_i$. The estimate for p is $\hat{p} = Z/N$. $E[Z] = Np$.

We want $\Pr(|\hat{p} - p| \geq \epsilon p) \leq \delta$. This is equivalent to $\Pr(|Z - Np| \geq$

ϵNp). Using Chernoff bounds (for small ϵ):

$$\Pr(|Z - Np| \geq \epsilon Np) \leq 2\exp\left(-\frac{\epsilon^2 Np}{3}\right).$$

We want this to be $\leq \delta$. Solving for N:

$$N \geq \frac{3}{p\epsilon^2}\log\left(\frac{2}{\delta}\right).$$

This is great if p is reasonably large (like estimating π, where $p = \Theta(1)$). However, for DNF counting, p might be tiny (e.g., 2^{-n}). In this case, N would be exponential. This is the "needle in a haystack" problem.

9.3 *Importance Sampling*

The key idea is that we can design our own "haystack" (the universe U).

Smart Monte Carlo Idea: Design a universe U such that:

1. We know the size $|U|$.
2. We can sample from U uniformly at random.
3. The target set $S \subseteq U$, and the ratio $p = |S|/|U|$ is "large" (i.e., $1/\text{poly}(n)$).

If these conditions are met, we can define an **unbiased estimator** for $|S|$. Sample x from U uniformly. Define the random variable $Y = |U| \cdot \mathbb{I}(x \in S)$.

$$E[Y] = |U| \cdot \Pr(x \in S) = |U| \cdot \frac{|S|}{|U|} = |S|.$$

We repeat this N times and average the results to reduce variance.

9.3.1 *Application: DNF Counting Revisited*

Let's apply this to DNF counting. Let $\varphi = C_1 \vee C_2 \vee \cdots \vee C_m$. Let A_i be the set of assignments that satisfy clause C_i. $|A_i|$ is easy to calculate: $|A_i| = 2^{n-\text{length}(C_i)}$.

Define the universe U as the disjoint union (multiset) of these sets:

$$U = \{(i, a) \mid i \in [m], a \in A_i\}.$$

The size $|U| = \sum_{i=1}^{m} |A_i|$ is known.

We want to count the total number of unique satisfying assignments $S_{SAT} = \bigcup A_i$. In U, an assignment a may appear multiple times (e.g., $(1, a), (5, a)$ if a satisfies C_1 and C_5).

We define the target set $S \subseteq U$ such that $|S| = |S_{SAT}|$. We select a "canonical" copy for each assignment.

$$S = \{(i, a) \in U \mid a \in A_i \text{ and } a \notin A_j \forall j < i\}.$$

Each satisfying assignment appears exactly once in S.

Now we analyze the ratio $p = |S|/|U|$.

Claim 9.4. $p = \frac{|S|}{|U|} \geq \frac{1}{m}$.

Proof. $|U|$ counts every satisfying assignment a exactly k_a times, where k_a is the number of clauses satisfied by a. Since there are m clauses, $1 \leq k_a \leq m$.

$$|U| = \sum_{a \in S_{SAT}} k_a \leq \sum_{a \in S_{SAT}} m = m \cdot |S_{SAT}| = m \cdot |S|.$$

□

This ratio is polynomially large.

We also need to be able to sample from U uniformly. This is done via **Importance Sampling**:

1. Pick index $i \in [m]$ with probability proportional to $|A_i|$. (i.e., $\Pr(\text{pick } i) = |A_i|/|U|$).
2. Pick an assignment a uniformly at random from A_i. (This is easy: fix the variables in C_i according to the clause, and choose the others randomly).

This generates a uniform sample (i, a) from U. We then check if $(i, a) \in S$.

Since $p \geq 1/m$, the number of samples required for an FPRAS is:

$$N = O\left(\frac{1}{p\epsilon^2} \log \frac{1}{\delta}\right) = O\left(\frac{m}{\epsilon^2} \log \frac{1}{\delta}\right).$$

This gives an FPRAS for DNF counting.

Remark 9.5. Suppose we have a randomized algorithm A such that for every $\varepsilon > 0$ and input of size n,

- A runs in time $\text{poly}(n, 1/\varepsilon)$, and
- with probability at least $p(n) \geq 1/\text{poly}(n)$, the output of A is a $(1 \pm \varepsilon)$-approximation to the desired quantity.

Then, by boosting the success probability of A via repetition and the Median-of-Means trick, we obtain an FPRAS: for any $\delta > 0$ there is a randomized algorithm that runs in time $\text{poly}(n, 1/\varepsilon, \log(1/\delta))$ and outputs a $(1 \pm \varepsilon)$-approximation with probability at least $1 - \delta$.

9.4 From Sampling to Counting

We saw that efficient sampling can lead to efficient approximate counting. Let's explore this connection for "self-reducible" problems, such as counting graph colorings.

Suppose we are given a graph $G = (V, E)$ and a number k. A proper k-coloring is a map $\chi : V \to [k]$ such that $\chi(u) \neq \chi(v)$ for all $(u, v) \in E$.

Assume we have a black-box sampler that can generate a proper k-coloring of G uniformly at random. How can we use this sampler to count the total number of k-colorings?

9.4.1 The Telescoping Product Idea

Let $\Omega(G)$ denote the set of proper k-colorings of G. We want to estimate $|\Omega(G)|$. Let $m = |E|$. Define a sequence of graphs starting from G and removing one edge at a time:

$$G = G_0 \supset G_1 \supset G_2 \supset \cdots \supset G_m = \text{Empty graph}.$$

We can write $|\Omega(G)|$ as a telescoping product:

$$|\Omega(G_0)| = \frac{|\Omega(G_0)|}{|\Omega(G_1)|} \cdot \frac{|\Omega(G_1)|}{|\Omega(G_2)|} \cdots \frac{|\Omega(G_{m-1})|}{|\Omega(G_m)|} \cdot |\Omega(G_m)|.$$

We know $|\Omega(G_m)| = k^n$, since for the empty graph, any assignment of colors is proper.

Let $p_i = \frac{|\Omega(G_i)|}{|\Omega(G_{i+1})|}$. If we can estimate each p_i up to a factor of $(1 \pm \epsilon')$, the product is accurate up to $(1 \pm \epsilon')^m$. If we set $\epsilon' = \epsilon/m$, this is approximately $(1 \pm \epsilon)$.

9.4.2 Estimating the Ratios

How do we estimate $p_i = \frac{|\Omega(G_i)|}{|\Omega(G_{i+1})|}$? Note that G_i has more constraints than G_{i+1}, so $\Omega(G_i) \subseteq \Omega(G_{i+1})$, and $0 < p_i \leq 1$.

We can use the Monte Carlo method.

1. Use the sampler to generate a uniform random coloring $\chi \in \Omega(G_{i+1})$.
2. Check if $\chi \in \Omega(G_i)$.

The probability of success is exactly p_i.

For this to be efficient, we need p_i to be large. Let Δ be the maximum degree of G.

Claim 9.6. If $k \geq \Delta + 2$, then $p_i = \frac{|\Omega(G_i)|}{|\Omega(G_{i+1})|} \geq \frac{1}{2}$.

Proof Idea. Let $G_i = G_{i+1} \cup \{u, v\}$. We want to show $|\Omega(G_{i+1})| \leq 2|\Omega(G_i)|$. Consider the "bad" colorings $S_{bad} = \Omega(G_{i+1}) \setminus \Omega(G_i)$. These are colorings where $\chi(u) = \chi(v)$. We want to show an injection from S_{bad} to $S_{good} = \Omega(G_i)$. If $\chi \in S_{bad}$, we can try to recolor v. Since $k \geq \Delta + 2$, there is always at least one color available for v that is different from $\chi(u)$ and all neighbors of v. This transforms χ into a coloring in S_{good}. A careful construction ensures this mapping can be made injective. □

If $p_i \geq 1/2$, the number of samples needed per step to achieve $(1 \pm \epsilon/m)$ accuracy with probability $(1 - \delta/m)$ (allowing a union bound over m steps) is:

$$N = O\left(\frac{1}{p_i(\epsilon/m)^2} \log \frac{m}{\delta}\right) = O\left(\frac{m^2}{\epsilon^2} \log \frac{m}{\delta}\right).$$

The total time is $m \cdot N = O(\frac{m^3}{\epsilon^2} \log \frac{m}{\delta})$.

This shows that if we can efficiently sample k-colorings (for $k \geq \Delta + 2$), we can efficiently count them. This is the idea of "reducing counting to sampling". This works for many "self-reducible" problems.

Remark 9.7. The reduction also works if the sampler is only approximately uniform. Furthermore, it is often possible to reduce sampling to counting as well. The picture for self-reducible problems is:

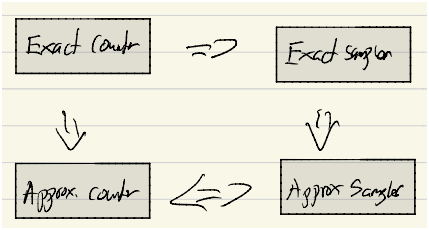

Figure 9.2: Approximate Counting and Approximate Sampling is "equivalent" for Self-Reducible Problems.

9.5 Markov Chain Monte Carlo (MCMC)

How can we generate these (approximately) uniform samples? A common technique is **Markov Chain Monte Carlo (MCMC)**.

The basic idea:

1. Set up a Markov Chain (MC) on the universe U (e.g., k-colorings).
2. Design the transitions such that the stationary distribution π^* is the desired distribution (e.g., uniform).
3. Ensure the chain is ergodic (irreducible and aperiodic), so the distribution at time t, $\pi^{(t)}$, converges to π^* as $t \to \infty$.

4. Simulate the MC for T steps. We need to determine how large T must be so that $\pi^{(T)}$ is "close enough" to π^*. We say the chain has "mixed".

9.5.1 Example: Glauber Dynamics for Colorings

Let's design an MCMC sampler for k-colorings of G. The state space is $\Omega(G)$.

Definition 9.8 (Glauber Dynamics)**.** Starting from a proper coloring χ_t:

1. Pick a vertex $v \in V$ uniformly at random.
2. Pick a color $c \in [k]$ uniformly at random.
3. Define χ' by setting $\chi'(v) = c$ and $\chi'(w) = \chi_t(w)$ for $w \neq v$.
4. If χ' is a proper coloring, set $\chi_{t+1} = \chi'$. Otherwise, set $\chi_{t+1} = \chi_t$.

We verify the properties of this chain.

Aperiodicity. There is a non-zero probability of staying at the current state (e.g., pick v and $c = \chi_t(v)$). So the chain is aperiodic.

Irreducibility. If $k \geq \Delta + 2$, the chain is irreducible.

Proof Sketch. We can get from any coloring χ to any other χ^*. We iteratively transform χ to χ^*. If $\chi(v) \neq \chi^*(v)$, we want to change the color of v. Since $k \geq \Delta + 2$, there is always a "free color" available, which allows us to move colors around until we can assign $\chi^*(v)$ to v while maintaining a proper coloring at all times. □

Stationary Distribution. Since the chain is ergodic, it has a unique stationary distribution π^*. We show it is uniform by checking the detailed balance equations. Let χ, χ' be two colorings that differ only at vertex v. The transition probability is $P_{\chi,\chi'} = \frac{1}{n} \cdot \frac{1}{k}$. Since $P_{\chi,\chi'} = P_{\chi',\chi}$ (the transitions are symmetric), the uniform distribution $\pi^*_\chi = 1/|\Omega(G)|$ satisfies detailed balance:

$$\pi^*_\chi P_{\chi,\chi'} = \frac{1}{|\Omega(G)|}\frac{1}{nk} = \pi^*_{\chi'} P_{\chi',\chi}.$$

Thus, the stationary distribution is uniform.

9.5.2 *Mixing Time and Total Variation Distance*

We want to know how quickly the chain converges. We measure the distance between distributions using Total Variation distance.

Definition 9.9 (Total Variation Distance)**.** The Total Variation (TV) distance between two probability distributions Π and $\mathcal{T}$ over a space Ω is:

$$||\Pi - \mathcal{T}||_{TV} := \frac{1}{2} \sum_{\omega \in \Omega} |\Pi(\omega) - \mathcal{T}(\omega)| = \max_{A \subseteq \Omega} |\Pi(A) - \mathcal{T}(A)|.$$

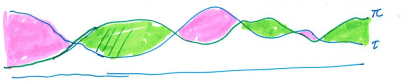

Figure 9.3: Illustration of TV distance.

The TV distance measures how distinguishable two distributions are. If $||\Pi - \mathcal{T}||_{TV} \leq \epsilon$, then for any event A, the probabilities under both distributions differ by at most ϵ.

Definition 9.10 (Mixing Time)**.** The mixing time $\tau(\epsilon)$ is the minimum time T such that for any starting distribution $\pi^{(0)}$,

$$||\pi^{(T)} - \pi^*||_{TV} \leq \epsilon.$$

There are several techniques to analyze mixing time (e.g., analyzing bottlenecks/expansion/spectral gap). Today we focus on **Coupling**.

9.6 *The Coupling Method*

Definition 9.11 (Coupling)**.** A coupling of a Markov Chain M_t (with state space Ω and transition matrix P) is another Markov Chain $Z_t = (X_t, Y_t)$ on $\Omega \times \Omega$ such that, when viewed marginally, both X_t and Y_t evolve according to M_t.

1. $\Pr(X_{t+1} = x' \mid Z_t = (x, y)) = P_{x,x'}$
2. $\Pr(Y_{t+1} = y' \mid Z_t = (x, y)) = P_{y,y'}$

The moves of X_t and Y_t can be correlated.

The idea is to design a coupling such that X_t and Y_t tend to move closer together, and once they meet ($X_t = Y_t$), they stay together ($X_{t+1} = Y_{t+1}$).

Theorem 9.12 (Coupling Theorem)**.** *Let the coupling time T_ϵ be the minimum time T such that for any starting states $X_0 = x, Y_0 = y$, $\Pr(X_T \neq Y_T) \leq \epsilon$. Then the mixing time is bounded by the coupling time: $\tau(\epsilon) \leq T_\epsilon$.*

Proof. Let $X_0 = x$ be arbitrary and let $Y_0 \sim \pi^*$. Then $Y_t \sim \pi^*$ for all t. Let $T = T_\epsilon$. For any event $A \subseteq \Omega$:

$$\begin{aligned}\Pr(X_T \in A) - \pi^*(A) &= \Pr(X_T \in A) - \Pr(Y_T \in A) \\ &\leq \Pr(X_T \in A \wedge Y_T \notin A) \\ &\leq \Pr(X_T \neq Y_T) \leq \epsilon.\end{aligned}$$

Similarly, $\pi^*(A) - \Pr(X_T \in A) \leq \epsilon$. Therefore, $||\pi^{(T)} - \pi^*||_{TV} = \max_A |\Pr(X_T \in A) - \pi^*(A)| \leq \epsilon$. □

Example 9.13 (Random Walk on the Hypercube)**.** Consider the space $\{0,1\}^n$. The Glauber dynamics is: Pick location $i \in [n]$ u.a.r. Pick bit $b \in \{0,1\}$ u.a.r. Set $X_i = b$. We couple (X_t, Y_t) by picking the **same** location i and the **same** bit b for both. If X_t, Y_t differ at i, they will agree at i after the update. The distance decreases. Once they meet, they stay together. The time until they meet is the time until every coordinate has been picked (Coupon Collector). The expected time is $O(n \log n)$. The coupling time is $O(n \log(n/\epsilon))$.

9.6.1 Coupling for Colorings

We now analyze the coupling time for the Glauber dynamics on colorings. We will show that if k is large enough, the chain exhibits "rapid mixing".

Theorem 9.14. *If the number of colors $k \geq 4\Delta + 1$, the mixing time of the Glauber dynamics is $O(nk \log(n/\epsilon))$.*

We use the **Synchronous Coupling** (as in the hypercube example):

Definition 9.15 (Synchronous Coupling for Glauber Dynamics)**.** Given (X_t, Y_t):

1. Pick the **same** vertex $v \in V$ u.a.r.
2. Pick the **same** color $c \in [k]$ u.a.r.
3. Update both X_t and Y_t using (v, c) according to the Glauber dynamics rules (accept if valid, reject otherwise).

We analyze the evolution of the Hamming distance: $d_t = \#\{v \in V \mid X_t(v) \neq Y_t(v)\}$.

We analyze the expected change in d_t in one step (Drift Analysis).

Good Moves (Convergence, decrease d_t). This occurs if we pick v where $X_t(v) \neq Y_t(v)$ (Prob d_t/n), and pick a color c valid for both chains. X_t blocks at most Δ colors at v. Y_t blocks at most Δ colors. At most 2Δ colors are blocked in total. So, at least $k - 2\Delta$ colors are valid for both.

$$\Pr(\text{Good}) \geq \frac{d_t}{n} \cdot \frac{k - 2\Delta}{k}.$$

Bad Moves (Divergence, increase d_t). This occurs if we pick v where $X_t(v) = Y_t(v)$, and pick c valid for one chain but not the other. Suppose c is valid for X_t but not Y_t. This means c matches the color of a neighbor w in Y_t, but not in X_t. Thus, $X_t(w) \neq Y_t(w)$. A bad move requires picking a vertex v adjacent to a disagreement, and picking one of the specific colors involved in the disagreement at the neighbor. There are d_t disagreements. Each can affect at most Δ neighbors. So there are at most $d_t\Delta$ such vertices v. For each such v, there are at most 2 colors (the colors of the disagreeing neighbor in X_t and Y_t) that can cause a bad move.

$$\Pr(\text{Bad}) \leq \frac{d_t\Delta}{n} \cdot \frac{2}{k} = \frac{2\Delta d_t}{nk}.$$

Drift Analysis. We calculate the expected drift $E[d_{t+1} - d_t \mid X_t, Y_t] = \Pr(\text{Bad}) - \Pr(\text{Good})$.

$$\begin{aligned} E[d_{t+1} - d_t] &\leq \frac{2\Delta d_t}{nk} - \frac{(k-2\Delta)d_t}{nk} \\ &= \frac{d_t}{nk}(2\Delta - (k-2\Delta)) = \frac{d_t}{nk}(4\Delta - k). \end{aligned}$$

If we assume $k \geq 4\Delta + 1$, then $4\Delta - k \leq -1$.

$$E[d_{t+1} - d_t] \leq -\frac{d_t}{nk}.$$

So, the distance contracts in expectation:

$$E[d_{t+1} \mid d_t] \leq d_t \left(1 - \frac{1}{nk}\right).$$

By induction (and the law of total expectation):

$$E[d_T] \leq d_0 \left(1 - \frac{1}{nk}\right)^T \leq n \left(1 - \frac{1}{nk}\right)^T.$$

We use the inequality $1 - x \leq e^{-x}$:

$$E[d_T] \leq n \cdot e^{-T/(nk)}.$$

We want $E[d_T] \leq \epsilon$. We choose $T = nk \ln(n/\epsilon)$.

$$E[d_T] \leq n \cdot e^{-\ln(n/\epsilon)} = n \cdot \frac{\epsilon}{n} = \epsilon.$$

Finally, we use Markov's inequality (First Moment Method) to bound the probability that the chains have not coupled:

$$\Pr(X_T \neq Y_T) = \Pr(d_T \geq 1) \leq E[d_T] \leq \epsilon.$$

The coupling time, and thus the mixing time, is $O(nk \log(n/\epsilon))$.

9.6.2 *Summary*

We showed that for $k \geq 4\Delta + 1$, we can sample k-colorings approximately uniformly in polynomial time using MCMC (Glauber dynamics).

Combining this with the reduction from counting to sampling (which required $k \geq \Delta + 2$), we obtain an FPRAS for counting k-colorings when $k \geq 4\Delta + 1$.

MCMC is a vast and active area of research (see notes by Eric Vigoda or Mark Jerrum) with many applications in computer science and statistical physics (e.g., Potts model, Hard-Core model (sampling independent sets), Ising model (sampling cuts)). Better results for coloring are known (e.g., $k > 11\Delta/6$).