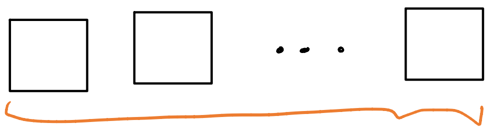
CS 263 : Counting & Sampling

Instructor (me)

CA

Nima Anari

Ryan Cooper



≃ 30 of you

- Aero & Astro
- Applied Physics
- Computer Science
- ICME
- Mathematics

undergrad +

masters +

Ph.D.

Logistics

Lecture: Mon, Wed 1:30pm - 2:50pm (best-effort recorded)

Website: cs263.stanford.edu

Office hours: Starts next week

Homework: 4 sets (20% each)

Final report: 20% of grade

- Solo or groups of 2
- Research: new progress on a problem relevant to the course

or

- Survey: choose ≥ 3 papers on a common problem/topic and survey them

Plan:

- What is "Counting & Sampling"?
- A bit of complexity theory

  don't worry, this is almost all the complexity theory you'll see in this class
- Approximate notions of counting & sampling
- First algorithms: Monte Carlo + Rejection Sampling

  if time

What is "Sampling & Counting"?

Distribution μ on large Ω ← finite but exp. large in most of this course

- Sampling: efficient alg. to produce sample $\omega \sim \mu$.
- Counting: compute $P_\mu[\text{event}]$ for various events of interest.

Example: #SAT

Input: $\Phi = (x_1 \vee \bar{x}_2 \vee x_3) \wedge (\bar{x}_1 \vee x_2 \vee x_4) \wedge \cdots$

Ω: $\{0,1\}^n$ ← assignments to n vars

μ: uniform over satisfying assignments

Why is it called Counting?

Let us compute $P_{x \sim \mu}[x_1 = 1] =$

$$\frac{\#\text{sat. assignments of } \Phi \text{ with } x_1 = 1}{\#\text{sat assignments of } \Phi}$$

so is this

this is a count

Clever observation:

numerator = # sat assignments to

$\Phi' := (\Phi \wedge x_1)$

- This sort of thing is called "self-reducibility"

will come back later

Formalism

Suppose μ is "unnormalized"

density : $\mu : \Omega \to \mathbb{R}_{\geq 0}$

w.r.t. an easy background measure on Ω

finite Ω usually has uniform background

- Sampling: Efficiently produce ω with $P[\omega] \propto \mu(\omega)$
- Counting: Compute the normalizing factor $\sum_{\omega \in \Omega} \mu(\omega)$

Partition Function

Standard Assumption:

μ is easy to compute for every point $\omega \in \Omega$.

Example. #SAT

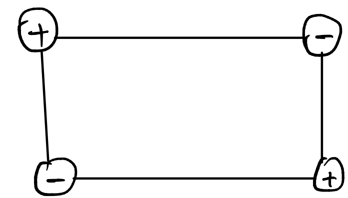
Input: $\Phi = (x_1 \vee \bar{x}_2 \vee x_3) \wedge (\bar{x}_1 \vee x_2 \vee x_4) \wedge \cdots$

$\Omega : \{0,1\}^n \leftarrow$ assignments

$\mu : \Omega \to \{0,1\}$

$\mu(x) = 1[x \text{ sats } \Phi]$

Example. Spin systems



graph $G = (V, E)$

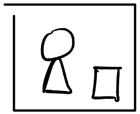
$\Omega : \underbrace{\{+,-\}}_{\text{could be larger domain}}{}^{V}$

$\mu(X) = \prod_{(u,v) \in E} \Phi(X_u, X_v)$

local interaction

Example. Generative ML models

Ω: all raw $n \times n$ images

μ:  $\longmapsto$ "realism of image"

⚠ We don't know μ. We learn something about it from data.

Score-based models:

learn approx. $\nabla \lg \mu$

x $\quad$ $x + \Delta x$

nearby points

$$\frac{\mu(x+\Delta x)}{\mu(x)} \simeq \exp(\nabla \lg \mu(x) \cdot \Delta x)$$

Bit of Complexity Theory

Example. Poly-time nondet. Turing machine M.

$M : (x, y) \longmapsto \{\text{Accept}, \text{Reject}\}$

our μ if Accept $=1$, Reject $=0$

input

witness/nondet choices

E.g.

M_{SAT}: (formula Φ, assignment x) $\longmapsto$
$\begin{cases} \text{Accept if } x \text{ sats } \Phi \\ \text{Reject o.w.} \end{cases}$

$NP = \{ x \longmapsto [\exists y : M(x,y)] \mid M \}$

$\#P = \{ x \longmapsto \#\{y : M(x,y)\} \mid M \}$

Every NP problem has a #P variant.

not unique

#P-complete: Every other #P prob. poly-time reduces to it.

Examples of #P problems:

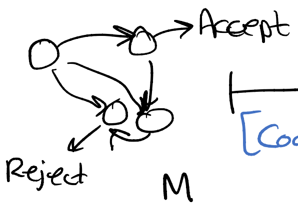
- #SAT
- # Hamiltonian Cycles
- # 3-colorings of Graph
- # Spanning Trees in Graph
- # Bipartite Perfect Matchings
- # Stable Matchings
- ...

Poll if time:

Which ones are #P-complete?

Reductions

All NP probs reduce to SAT.



$\longmapsto (x_1 \vee \bar{x}_2 \vee \cdots) \wedge \cdots$

[Cook-Levin]

This reduction is parsimonious.

accepting paths $\xleftrightarrow{\text{one-one}}$ sat assignments

m-to-n also called parsimonious

Corollary: #SAT is #P-complete.

In fact, all natural NP-complete problems we know have parsimonious reductions:

#3-colorings #Hamiltonian paths

...

Open problem: Do all NP-complete problems have a #P-complete variant?

#P-complete: really, really hard

Obvious: NP-hard

[Toda]: Poly Hierarchy $\subseteq P^{\#P}$

$\forall x \exists y \forall z \exists \cdots M(x, y, z, \ldots)$

Counting variants of NP-complete problems are hopeless :/

But even P problems could become #P-complete. :(

Example. Count sat assignments to DNF formula: $(x_1 \wedge \bar{x}_2 \wedge x_3) \vee (\cdots) \vee \cdots$

Proof: $\#DNF = 2^n - \#CNF$ $\square$

Example [Valiant '79]. Given bipartite graph, counting perfect matchings is #P-complete!

Note: these reductions are not parsimonious!

Casual observation: Efficient counting known for only a handful of interesting problems.

- #spanning trees
- # planar perfect matchings
- # directed Eulerian circuits
- Determinantal point processes

Common feature: all reducible to matrix determinants.

All hope is lost?

Approximation to the rescue

- Approximate counting:
  Output Z with count $\in [Z, (1+\varepsilon)Z]$
- Fully Poly-Time Approx. Scheme:
  Above with runtime $\text{poly}(n, \frac{1}{\varepsilon})$
  input size
  Abbr: FPTAS
- Fully Poly-Time Rand. Approx. Scheme:
  Above but with randomness and $\frac{2}{3}$ chance of success.
  Abbr: FPRAS

HW: $\frac{2}{3}$ can be replaced with $1-\delta$ and runtime $\leq \text{poly}(n, \frac{1}{\varepsilon}, \lg \frac{1}{\delta})$.

Question: Why all ε? Why not 100-approx?
Answer: Approx counting is all-or-nothing.

Example: #SAT

Suppose $f(n)$-approx alg A.

Give A $\Phi^{(1)} \wedge \Phi^{(2)} \wedge \cdots \wedge \Phi^{(t)}$ for disjoint copies of Φ.

$$\text{output} \overset{f(nt)}{\simeq} \#SAT(\Phi)^t$$

$$\Downarrow$$

$$\sqrt[t]{\text{output}} \overset{f(nt)^{\frac{1}{t}}}{\simeq} \#SAT(\Phi)$$

Say for $f(n) = 2^{n^{0.99}}$ we have

$f(nt)^{\frac{1}{t}} = 2^{n^{0.99}/t^{0.01}}$, so let

$t = (\frac{n}{\varepsilon})^{100} \Rightarrow f(nt)^{\frac{1}{t}} \leq 2^{\varepsilon} \simeq 1+\varepsilon$.

Applies to "tensorizable" problems

[Jerrum-Sinclair] Any "self-reducible" problem with poly(n)-approx alg has an FPRAS.

We will see this later in the course.

- Approximate Sampling:

For dists ν, μ on Ω define

$$d_{TV}(\nu,\mu) = \max\{P_\nu[B] - P_\mu[B] \mid \text{event } B\}$$
$$= \frac{1}{2}\sum_{\omega\in\Omega} |\mu(\omega) - \nu(\omega)|$$

because #P

- Fully Poly-Time Approx. Uniform Sampler.

For δ output ω in time $\mathrm{poly}(n, \lg\frac{1}{\delta})$ s.t. $d_{TV}(\text{dist of } \omega, \text{normalized } \mu) \leq \delta$.

Abbr: FPAUS

For "self-reducible" problems

$$\text{Counting} \equiv \text{Sampling}$$

$$\begin{tikzcd}
\text{Exact Counting} \arrow[r] \arrow[d] & \text{Approx Counting (FPRAS)} \arrow[dl, dashed, "\text{if FPTAS}"'] \arrow[d, leftrightarrow, "\text{[Jerrum-Valiant-Vazirani]}"] \\
\text{Exact Sampling} \arrow[r] & \text{Approx Sampling (FPAUS)}
\end{tikzcd}$$

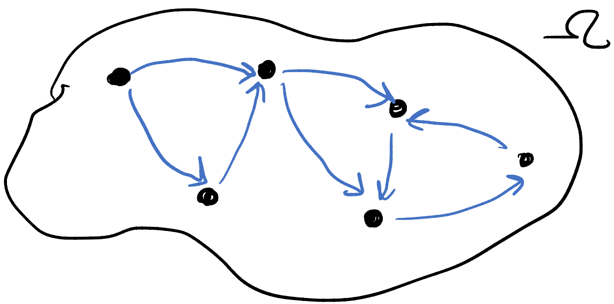
Arrow: poly-time reduction.

We will prove these next class.

Aside: Most important direction is

Approx Sampler $\longrightarrow$ Approx Counter

A lot of this class will be about sampling via Markov Chains.



$$x_0 \xrightarrow{\text{rand jump}} x_1 \xrightarrow{\text{rand jump}} x_2 \longrightarrow \cdots \longrightarrow x_t$$

hope this is close to μ

#DNF

Input: DNF formula

$$(x_1 \wedge \bar{x}_2 \wedge x_3) \vee \cdots$$

Can we approx. sample/count satisfying assignments?

Attempt #1 (naive Monte Carlo):

Sampler:

- Sample u.r. $x \sim \{0,1\}^n$
- if x is sat
  - Accept and return x
- else
  - Reject and try again

This is an instance of rejection sampling.

Rejection Sampling

We can sample from ν but want to sample $\propto \mu$.

Loop:

- Sample $x \sim \nu$
- Accept w.p. $\frac{C \cdot \mu(x)}{\nu(x)}$ choose so prob ≤ 1
- If reject, loop again ...

Good: Output $\sim$ normalized μ

Bad: Can take a long time:

$$P[\text{Accept}] = C \cdot \left(\sum_x \mu(x)\right)$$

For DNFs, $C = \frac{1}{2^n}$, so $P[\text{Accept}] = \frac{\#\text{sat}}{2^n}$

This can be small: $(x_1 \wedge x_2 \wedge \cdots \wedge x_n)$

Attempt #2 [Karp-Luby]:

$$\Phi = C_1 \vee C_2 \vee \cdots \vee C_m$$

each clause with k vars has 2^{n-k} sat assignments

- $A_i = \{$sat assignments of clause $i\}$
- We want to sample from $A_1 \cup A_2 \cup \cdots \cup A_m$.

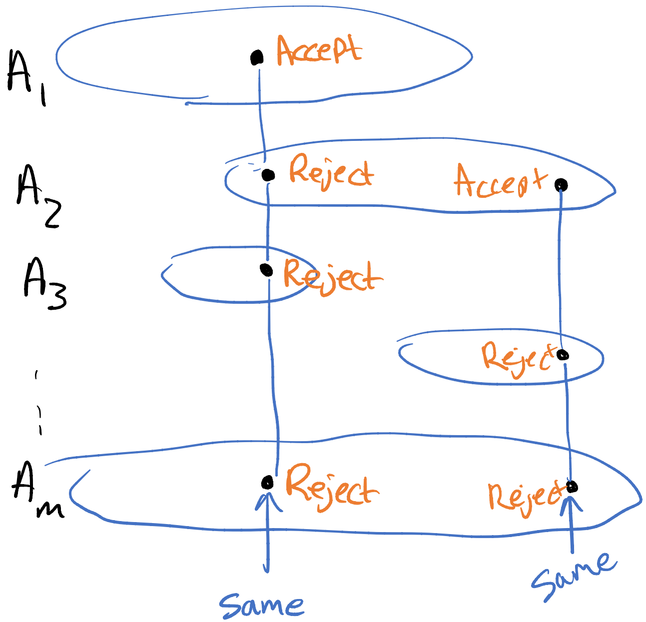
Main Idea: Sample from

$$A_1 \sqcup A_2 \sqcup \cdots \sqcup A_m$$

disjoint union

and rejection-sample it into $A_1 \cup \cdots \cup A_m$!

$$|A_1 \sqcup \cdots \sqcup A_m| = \sum |A_i| \leq m \cdot \max |A_i| \leq m \cdot |A_1 \cup \cdots \cup A_m|$$



Alg: Loop

- Sample $x \in A_1 \sqcup \cdots \sqcup A_m$
- If x sampled from A_i and no A_j with $j < i$ has x Accept
- Otherwise, try again.

$$\Pr[\text{Accept}] \geq \frac{1}{m} \Rightarrow \text{expected loops} = O(m)$$

If we want to stop early we can pay $\left(1 - \frac{1}{m}\right)^t$ in d_{TV} and output garbage if we reject for t rounds

$$t \geq m \lg \frac{1}{\delta} \Rightarrow \left(1 - \frac{1}{m}\right)^t \leq e^{-\frac{t}{m}} \leq \delta.$$

Approx Counting of DNFs

Idea (Naive Monte Carlo):

$$\Pr[\text{Accept}] = \frac{|A_1 \cup \cdots \cup A_m|}{|A_1 \sqcup \cdots \sqcup A_m|}$$

estimate this

we know this

multiply together.

Estimation: try t times

$$\frac{1[\text{Accept in try } 1] + \cdots + 1[\text{Accept in try } t]}{t}$$

Thm: When $t > \dfrac{3}{\varepsilon^2 P[\text{accept}]}$ this gives a $(1+\varepsilon)$ approx to $P[\text{accept}]$ w. prob $\geq \frac{2}{3}$.

Open Problem: Is there FPTAS?

Best known result due to [Gopalan–Meka–Reingold] has time $\simeq n^{\tilde{O}(\lg \lg n)}$

Proof: $X_i = 1[\text{accept in try } i]$

$p = P[\text{accept}]$

$$X = \frac{X_1 + \cdots + X_t}{t}$$

- $E[X] = p$
- $\mathrm{Var}(X_i) = p(1-p) \leq p$
- $\mathrm{Var}(X) \leq \frac{p}{t}$

By Chebyshev's ineq:

$$\Pr[X \notin [p - \varepsilon p, p + \varepsilon p]] \leq \frac{p/t}{(\varepsilon p)^2}$$

$$= \frac{1}{t \cdot p \cdot \varepsilon^2}$$

So if $t > 3/\varepsilon^2 p$ then

$$P[\text{failure}] < \frac{1}{3}.$$