6.5420 Lecture 1: Randomness & Computation

Lecturer: Ronitt Rubinfeld

TA: Lily Chung

Course website:

https://people.csail.mit.edu/ronitt/COURSE/S26/

Today:

- Course Overview
- Polynomial Identity Testing

  applications to:

  - "person on the moon"
  - bipartite matching

What is this course about?

- How can randomness help in computation?

  - algorithm design
    simpler, faster, new problems
  - show existence of combinatorial objects
    good solutions, codes, nice graphs
  - easy to verify proofs
    interactive proofs, PCPs
  - distributed algorithms
  - learning algorithms
  - testing algorithms

- A theme:

Randomness vs. Predictability

- computational learning theory (predictable)
  - learning vs. predictability
  - learning const depth ckts, decision trees, noisy parity fctns
  - weak learning + BOOSTING

- Pseudorandomness (unpredictable)
  - Pseudorandom generators
  - derandomization
  - recycling randomness

. Tools:

Fourier representation

random walks / Markov chains

algebraic techniques

probabilistic proofs

Lovasz Local Lemma

graph expansion, extractors

Szemeredi Regularity Lemma

Administratrivia

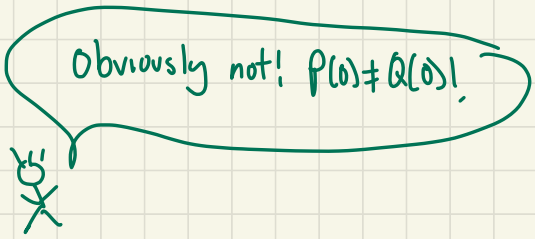
(see course handout)

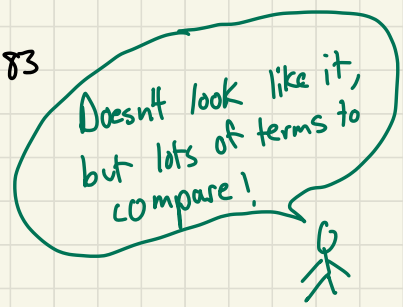
Polynomial Identity Testing

Is $P(x) = (x+1)^2$ the same as $Q(x) = x^2+2x+1$?

What about $P(x) = (x+3)^{38}(x-4)^{83}$

& $Q(x) = (x-4)^{38}(x+3)^{83}$





Problem: given 2 polynomials P, Q

is $P \equiv Q$?

i.e. is $P(x) = Q(x)\ \forall x$?

Problem': given polynomial R

is $R \equiv 0$?

i.e. is $R(x) = 0\ \forall x$?

Problems are equivalent!!

Let $R(x) = P(x) - Q(x)$

then $R \equiv 0$ iff $P \equiv Q$

(Fundamental Thm of Algebra)

Fact: If $R \not\equiv 0$ has degree $\leq d$ then

R has at most d roots *

(recall: a "root" is x s.t. $R(x)=0$)

Algorithm for deciding whether $R \equiv 0$: (Promised that $\deg R \leq d$)

> pick $d+1$ distinct inputs $x_1 \cdots x_{d+1}$
>
> if $\forall i \; R(x_i) = 0$ output "$R \equiv 0$"
>
> else $(\exists i \text{ s.t. } R(x_i) \neq 0)$ output "$R \not\equiv 0$"

How many evaluations of R?

$d+1$

* this is true over any field

$\mathbb{Z}_q$, mod q, ...

↑ any prime $> d$

Faster randomized algorithm for deciding whether $R \equiv 0$:

Pick $2d$ distinct inputs $x_1, \dots x_{2d}$

Do k times:
  Pick $i \in [2d]$, if $R(x_i) \neq 0$ output "$R \not\equiv 0$"

Output "$R \equiv 0$"

Behavior:

if $R \equiv 0$, $\forall x_i \; R(x_i) = 0$ so always outputs "$R \equiv 0$"

if $R \not\equiv 0$, $\Pr_{i \in [2d]}[R(x_i) = 0] \leq \frac{\#\text{ roots}}{2d} \leq \frac{1}{2}$

$\Pr[\text{err}] = \Pr[\text{choose root in all } k \text{ iterations}] \leq \frac{1}{2^k}$

$\Rightarrow \Pr[\text{correct}] = \Pr[\text{output "}R \not\equiv 0\text{"}] \geq 1 - \frac{1}{2^k}$

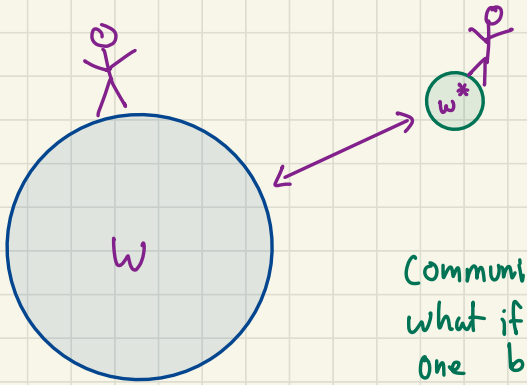
↑ Pset 1 practice problem 4

If you are willing to tolerate prob of error $\leq \delta$,
pick $k = \log 1/\delta$

How many evaluations? $O(\log \frac{1}{\delta})$

WOW! doesn't depend on degree, just δ

Application: "Person-on-the-moon"



Question: Is $w = w^*$?

Communication is expensive!! what if they differ in only one bit?

$w = w_0 \dots w_n$ ($n+1$ bit string)

naive solution: send $n+1$ bits

Let $P(x) = w_n \cdot x^n + w_{n-1} \cdot x^{n-1} + \dots + w_1 x + w_0$

$P^*(x) = w_n^* x^n + w_{n-1}^* x^{n-1} + \dots + w_1^* x + w_0^*$

$w = w^* \iff P \equiv P^*$ for P, P^* of degree n

Instead of sending full description of w,

- earth person picks random $r_1 \dots r_k$ & sends
$$(r_1, P(r_1))\ (r_2, P(r_2)), \dots (r_k, P(r_k))$$
- person on moon then
  1) computes $P^*(r_1), P^*(r_2), \dots, P^*(r_k)$
  2) checks $P(r_i) = P^*(r_i)\ \forall\ i \in [k]$

} mod q

Wait!!! x^n needs $\log(x^n) = n \log x$ bits to write down

Too big!!!

Solution: compute mod q

where q is a prime $> 2n$

Fact: q doesn't need to be bigger than $c \cdot n$ so only need $O(\log n)$ bits per number

How many bits of communication?

r_i's in $[2n] \Rightarrow O(\log n)$ bits to specify

$P(r_i)$'s in $[q] \Rightarrow O(\log n)$ " " "

$k = O(\log 1/\delta)$

total communication bits: $O(\log n \cdot \log \frac{1}{\delta})$

Behavior:

if $P \equiv P^*$, $P(r_i) = P^*(r_i)$ $\forall i$

if $P \not\equiv P^*$, $P(r_i) = P^*(r_i)$ with prob $\leq 1/2$

Multivariate Polynomial Identity Testing

Test if $R(x_1, x_2, \ldots, x_n) \equiv 0$

Total degree: $\max_{s \in \text{terms}} \left(\text{sum of degrees of } x_i\text{'s in term } s \right)$

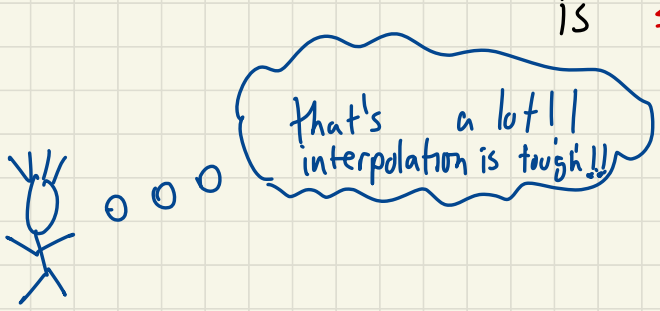
e.g. $\underbrace{2xy}_{\deg 2} + \underbrace{3z^3}_{\deg 3} + \underbrace{4xyz^2}_{\deg 4}$ total deg 4

difficulty 1: $R \not\equiv 0$ can have infinitely many roots

e.g. $R(x,y) = x \cdot y$

$R_2(x,y) = x - y$

difficulty 2: # terms in total degree d poly is $\leq \binom{n}{d}$



Good news!

[Schwartz-Zippel-DeMillo Lipton]

For R of total degree d s.t. $R \not\equiv 0$:

Given S containing $2d$ elements

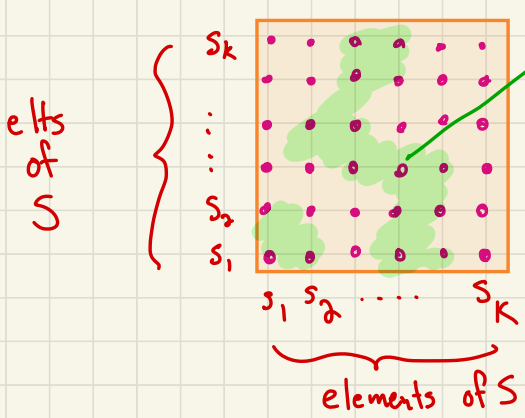
Pick $x_i \in_R S \quad \forall i$ ← Pick $x_1 \dots x_n$ from "n-dim cube"

Then $\Pr[R(x_1 \dots x_n) = 0] \leq \frac{d}{|S|}$

Proof induction on d

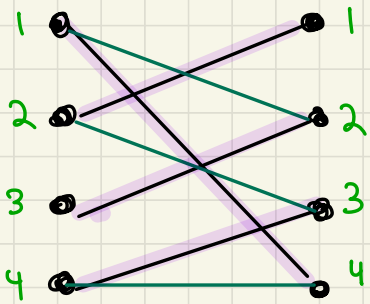
e.g. for $n=2$

$\Pr[R(s_i, s_k) = 0] \leq \frac{d}{|S|}$



Application:

Bipartite Perfect Matching



$|L| = n$ $\quad$ $|R| = n$

Matching: $M \subseteq E$

no two edges share endpt

Perfect Matching:

$|M| = n$

(all nodes get matched)

↑ Can solve in polytime via flows

Today: another approach!

def S_n: all "permutations" of 1 to n

e.g. $\sigma : [n] \to [n]$ & σ is a bijection

(for $i \neq j$, $S_n[i] \neq S_n[j]$

for all $k \in [n]$, $\exists\, i$ s.t. $S_n[i] = k$)

a few examples of bijections:

note that this is the purple matching in our graph

nonexample:

i	$\sigma_1(i)$	$\sigma_2(i)$	$\sigma_3(i)$	$\phi(i)$
1	1	4	2	2
2	2	1	3	3
3	3	2	4	2
4	4	3	1	4

these two are not matchings in our example

Define special matrix for G:

$$a_{ij} = \begin{cases} x_{ij} & \text{if } (ij) \in E \\ 0 & \text{o.w.} \end{cases}$$

$$A = \left[a_{ij} \right]$$

in example:

$$A = \begin{bmatrix} 0 & x_{12} & 0 & x_{14} \\ x_{21} & 0 & x_{23} & 0 \\ 0 & x_{32} & 0 & 0 \\ 0 & 0 & x_{43} & x_{44} \end{bmatrix}$$

notoriously hard to compute

→ permanent $\quad \text{perm}(A) = \sum_{\sigma \in S_n} \prod_{i=1}^{n} a_{i\,\sigma(i)}$

→ determinant $\quad \det(A) = \sum_{\sigma \in S_n} \text{sgn}(\sigma) \prod_{i=1}^{n} a_{i\,\sigma(i)}$

→ easy?

Note that: permutation σ of $[n]$ corresponds to perfect matching M in our graph

$\Leftrightarrow$

$\forall i$, $(i, \sigma(i))$ is edge in matching

main insight: For given permutation σ:

term drops out if not perfect matching $\rightarrow$ $\prod_{i=1}^{n} a_{i,\sigma(i)}$ will be 0 if even one of $(i, \sigma(i)) \notin E$

so $\prod_{i=1}^{n} a_{i,\sigma(i)} \neq 0$ iff σ is a perfect matching

In example:

$\text{perm}(A_G) = x_{21} \cdot x_{32} \cdot x_{43} \cdot x_{14}$

$\det(A_G) = \text{sign}(\sigma_2) \cdot x_{21} \cdot x_{32} \cdot x_{43} \cdot x_{14}$

} all other terms = 0

so { $\text{Perm}[A] \neq 0$ / $\det[A] \neq 0$ } iff $\exists$ some σ which is a matching

(some term remains)

- Permanent is hard to compute
- Can compute Det of matrix of integers in poly time, but this a matrix containing variables

Det[A] is a polynomial! ← and a very BIG one!

n^2 vars (1 for each edge)

total degree n

terms $n!$ ← huge!!

Luckily, we don't need to compute Det[A]

We just need to decide if it is $\equiv 0$

Algorithm: Test $\text{Det}[A] \not\equiv 0$ via Schwarz-Zippel-Demillo-Lipton

use $|S| = 2n$