**CS760 Topics in Computational Complexity** **2024-25 Sem I**

Lecture 18: 08-10-2024

*Scribe: Antriksh Punia, Anilesh Bansal* *Lecturer: Rohit Gurjar*

# 1 PIT (Polynomial Identity Testing )

Let $f(x_1, x_2, \dots, x_n)$ be a non-zero polynomial of degree $d$. Let $a_1, a_2, \dots, a_n \in S$ be uniformly randomly and independently chosen. Then,

$$\Pr[f(a_1, a_2 ... a_n) = 0] \le \frac{d}{|S|}.$$

**A variation: Individual degree $\le \delta$**

$$\Pr[f(a_1, a_2 ... a_n) \ne 0] \ge \left(1 - \frac{\delta}{|S|}\right)^n$$

**Note:** Both can be proved using induction on number of variables however we will be exploring a different approach

## 1.1 Proof by Induction

Induction on $n$ - the number of variables. Base case $n = 1$ is trivial, because a univariate polynomial of degree $d$ has at most $d$ roots. Now we assume the theorem is true for polynomials with $n-1$ variables, and we prove it for those with $n$ variables. The main idea is to obtain polynomials with fewer variables by factoring out the variable $x_1$ from $f$. Let $k$ be the largest power of $x_1$ appearing in any monomial of $f$. Then, we can write

$$f(x_1, x_2, \dots, x_n) = \sum_{i=0}^{k} x_1^i \cdot p_i(x_2, \dots, x_n)$$

Consider the event $E$ that $p_k(a_2, \dots, a_n) = 0$. Let $\neg E$ be the event that $E$ does not happen. Notice that $p_k$ has a degree of atmost $d - k$.

$$\begin{aligned}
\Pr(f(x_1, x_2, \dots, x_n) = 0) &= \Pr(f(x_1, x_2, \dots, x_n) = 0 \mid E) \cdot \Pr(E) + \Pr(f(x_1, x_2, \dots, x_n) = 0 \mid \neg E) \cdot \Pr(\neg E) \\
&\le \Pr(E) + \Pr(f(x_1, x_2, \dots, x_n) = 0 \mid \neg E) \\
&= \frac{d-k}{|S|} + \frac{k}{|S|} = \frac{d}{|S|}. \qquad (1)
\end{aligned}$$

where the first term follows from induction hypothesis and the second is true because once $(a_2, \cdots, a_n)$ are chosen and fixed, $f$ can be viewed as a single variable polynomial in $x_1$ with degree $k$.

## 1.2 Another proof

**Argument 1:**

We can try to argue as follows. For any choice of $a_2, a_3, \dots, a_n$, the polynomial $f(x_1, a_2, a_3, \dots, a_n)$ is a polynomial of degree at most $d$. Hence, it has at most $d$ roots (if it's a nonzero polynomial). There are $|S|^{n-1}$ choices for $a_2, a_3, \dots, a_n$. If for each choice we had at most $d$ roots, then the total number of roots will be $\le d|S|^{n-1}$ as required.

However this argument doesn't work because it is possible that for some $(a_2, a_3, ... a_n) \in |S|^{n-1}$ we have $f(a_1, a_2, ... a_n) = 0$ for all possible $a_1$. We will slightly modify this argument to make it work.

- Argument 1 suggested $|S|^{n-1}$ lines each with max $d$ points but this doesn't work as some lines have all points as root.
- So instead of just considering the lines $(x_2, \ldots, x_n) = C$ for some C, we consider $|S|^{n-1}$ lines along a particular direction and hope that for all these lines, the argument that at most $d$ of the points of them are roots hold.

We will be able to show this only when size of $S$ is a prime number. For convenience, we will work with the set $S = \mathbb{Z}_p$ for some prime $p$. This means all arithmetic will be modulo prime $p$. Note that the fact that a univariate degree $d$ polynomial has at most $d$ roots, still holds over $\mathbb{Z}_p$ (this is not obvious, it holds because $\mathbb{Z}_p$ is a field).

We choose random $a_1, a_2, \ldots, a_n \in \{0, 1, \ldots, p-1\}$ and need to show that

$$\mathbb{P}[f(a_1, a_2, \ldots, a_n) \equiv 0 \pmod{p}] \leq \frac{d}{p}.$$

As an example, consider the case with $p = 5$ and

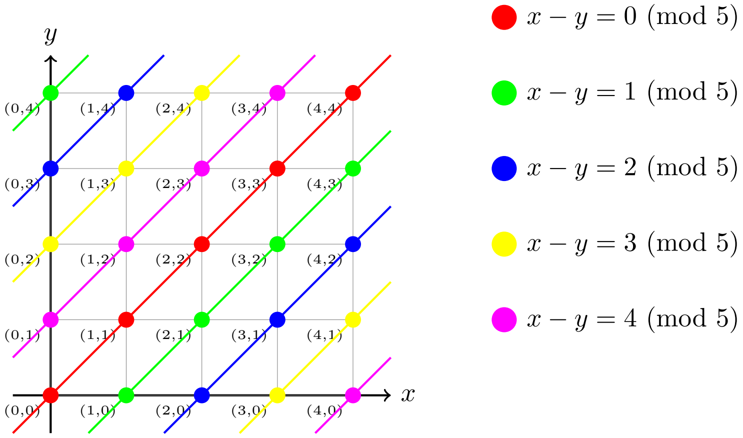
$$f(x_1, x_2) = x_1$$

If we considered the lines $x_1 = 0, 1, 2, 3, 4 \pmod{5}$, then the above argument wouldn't hold since the line $x_1 = 0$ has 5 roots.

But instead if we consider the lines $x_1 - x_2 = 0, 1, 2, 3, 4 \pmod{5}$, it can be easily seen that along each of the lines, at most $d = 1$ roots would occur.

See figure for explanation. Notice that there are 5 lines and each line contains exactly 5 points. This is true for all other directions, like say $2x - 3y = 0, 1, 2, 3, 4 \pmod{p}$.

Next we'll show that there exists at least one choice of the direction such that the above argument holds.



### Line

A line is defined as

$$L_{a,b} = \{a + tb : t \in \mathbb{F}_p\} \text{ where } a, b \in \mathbb{F}_p^n.$$

- Along this line, $f(a + tb)$ behaves like a function of one variable $t$, in fact, a polynomial of degree at most $d$. Therefore, if $f(a+tb)$ is a non-zero polynomial, the number of roots of $f$ on $L_{a,b}$ is at most $d$.
- We need to demonstrate that there exists a choice of $b$ (i.e., a direction) such that for every $a$, $f(a+tb)$ is a non-zero polynomial.
- Let $f_d$ be the degree $d$ part of the polynomial $f$. Observe that the coefficient of $t^d$ in $f(a + tb)$ is nothing but $f_d(b)$. That is, $a$ does not affect the coefficient of $t^d$ in $f(a + tb)$.

- We can choose $b$ such that $f_d(b) \neq 0$. This implies that for all $a$, $f(a + tb)$ has a non-zero leading coefficient, thus making the polynomial non-zero. We still need to prove the existence of such a direction $b$.

**Claim 18.1.** *For any degree-$d$, $n$-variate, nonzero polynomial $g(x_1, x_2, \ldots, x_n)$, if $p > d$, then $\exists\, b \in \mathbb{F}_p^n$ such that $g(b) \not\equiv 0 \pmod{p}$.*

*Proof.* We will prove this using induction on $n$, number of variables. It is clearly true for $n = 1$. Let us first write the polynomial $g$ as $x_1^d \cdot g_d + x_1^{d-1} \cdot g_{d-1} + \cdots + x_1^0 \cdot g_0$. Here each $g_j$ is a polynomial (possibly zero polynomial) in variables $x_2, \ldots, x_n$ and has degree at most $d$. Clearly, there exists at least one $0 \le j \le d$, for which polynomial $g_j$ is a nonzero polynomial. By induction hypothesis, there is a point $(b_2, \ldots, b_n) \in \mathbb{F}_p^{n-1}$ such that $g_j(b_2, \ldots, b_n) \not\equiv 0 \pmod{p}$. Now, we define polynomial $h(x_1)$ as

$$h(x_1) := g(x_1, b_2, \ldots, b_n) = x_1^d \cdot g_d(b_2, \ldots, b_n) + \cdots + x_1^0 \cdot g_0(b_2, \ldots, b_n).$$

We know that $h(x_1)$ is a nonzero polynomial (because $g_j(b_2, \ldots, b_n)$ is nonzero). Hence, there must be a choice $b_1 \in \mathbb{F}_p$ such that $h(b_1) \not\equiv 0 \pmod{p}$. It immediately follows that $g(b_1, b_2, \ldots, b_n) \not\equiv 0 \pmod{p}$. □

# 2 Some other problems

## PIT Problem

The Polynomial Identity Testing (PIT) problem involves determining whether a given polynomial is identically zero. A common instance of this problem is when the polynomial is expressed as the determinant of a matrix where the entries are linear polynomials (degree 1). Specifically, we are asked to check whether:

$$\det \begin{pmatrix} 2x_1 + 3x_2 & \cdots & \cdots \\ \cdots & 1 - 5x_3 & \cdots \\ \cdots & \cdots & \cdots \end{pmatrix}_{d \times d} \overset{?}{=} 0$$

is identically zero.

**Randomized Algorithm for PIT:** One simple approach is to substitute random values for $(x_1, x_2, \ldots, x_n)$ and check whether the determinant evaluates to zero. This is based on the idea that if the polynomial is non-zero, it is unlikely to vanish at a random point. This is what we proved above.

It remains an open question whether there exists a deterministic algorithm to solve this problem efficiently. However, it is believed that BPP=P, and if that is true then such a deterministic algorithm should exist. We will explore this topic further in the next lecture.

## BPP problems not known to be in P

Given a bipartite graph $G$ with edges colored either red or blue, the question is whether there exists a perfect matching that includes exactly $k$ red edges. This problem is known to have a simple randomized algorithm based on the determinant. However we don't know the answers to the following.

- Is it in P?
- Is it in coNP?

Another such problem is approximately counting the number of satisfying assignments for a given DNF formula. It is a search problem, which is known to have a randomized polynomial time algorithm. However no deterministic polynomial time algorithm is known. In fact, we don't even know if the decision problem – whether the number of satisfying assignments is at most a given threshold $T$ – is in NP or coNP.

### BPP problems not known to be in RP

Recall that RP corresponds to randomized algorithms with only one sided error, while BPP allows to make errors for both yes and no instances. The PIT example we have seen has an algorithm with only one sided error. Are there examples, which are known to be in BPP, but not in RP? Here is a somewhat unnatural example. Given three multivariate polynomials $C_1, C_2, C_3$, determine whether exactly two out of three given polynomials $C_1, C_2, C_3$ are equal. These polynomials are in some succinct representation, for example, as the determinant of a matrix.

# 3 BPP-Complete

It is relatively straightforward to construct $NP$-complete problems. For example:

{Given a nondeterministic Turing machine $M$, an input $x$ and $1^n$, does $M$ accept $x$ within $n$ steps?}

This problem, often referred to as the bounded halting problem, is a typical $NP$-complete problem, as it captures the essence of nondeterministic computation and can be verified in polynomial time.

In contrast, there is no such BPP-complete problem that emerges naturally from the definition.

# 4 Error reduction

Below we show that if we have a randomized algorithm for a problem that is correct with probability a little more than 1/2, then we can get another algorithm that will be correct with probability almost equal to 1. The running time will increase by a polynomial factor.

**Claim 18.2.** *Let $L \subseteq \{0,1\}^*$ be a language and suppose that there exists a polynomial-time probabilistic Turing Machine $M$ such that for every $x \in \{0,1\}^*, \Pr[M(x) = L(x)] \geq \frac{1}{2} + |x|^{-c}$.*

*Then for every constant $d > 0$ there exists a polynomial-time PTM $M'$ such that for every $x \in \{0,1\}^*$,*

$$\Pr[M'(x) = L(x)] \geq 1 - 2^{-|x|^d}.$$

*Proof.* Construct the PTM $M'$ as follows - on every input $x \in \{0,1\}^*$, run $M(x)$ for $k$ times obtaining $k$ outputs $y_1, \cdots, y_k \in \{0,1\}^k$, where $k = 4|x|^{2c+d}$. If the majority of these values are 1 then accept, otherwise reject.

Define for every $i \in [k]$ the random variable $X_i$ to equal 1 if $y_i = L(x)$ and to equal 0 otherwise. Note that $X_1, ..., X_k$ are independent Boolean random variables with $\mathbb{E}[X_i] = \mathbb{P}[X_i = 1] \geq \frac{1}{2} + n^{-c}$ (where $n = |x|$).

By Chernoff Bounds we know that if $X_1, \cdots, X_k$ be independent identically distributed Boolean random variables, with $\mathbb{P}[X_i = 1] = p$ for every $1 \leq i \leq k$. Let $\delta \in (0,1)$. Then,

$$\Pr\left[\sum_{i=1}^{k} X_i \leq (1-\delta)pk\right] \leq e^{-\delta^2 pk/2}.$$

In our case $p = \frac{1}{2} + n^{-c}$, and plugging in $\delta = n^{-c}$, the probability of getting wrong answer is bounded by

$$Pr[\frac{1}{k}\sum_{i=1}^{k} X_i \leq 1/2] \leq e^{-n^{-2c} \cdot 4n^{2c+d}/4} \leq e^{-n^d} < 2^{-n^d}.$$

□

## 4.1 BPP ⊂ P/poly

The idea is to take the random string that gives the correct answer as an advice. However, the same random string may not give correct answer for every possible input of a particular size. We can get around this problem by above probability amplification. We saw that if

$$Pr[M(x,r) \neq L(x)] \leq \frac{1}{2} - |x|^{-c}$$

then we can just repeat the algorithm $M$ $k$ times using independent random bits and output the majority as the result to get a new algorithm that succeeds with overwhelming probability. Using this, we can argue that there exists one choice of random string that works for all possible inputs of a particular size.

Suppose we have an algorithm that on input size $n$ succeeds with probability at least $1 - 2^{-n^2}$. In other words, given an input $x$ of size $n$, if we choose random string $r$ randomly, with probability at most $2^{-n^2}$, it is a wrong choice for input $x$. Using union bound, we can argue that a randomly chosen string $r$ is a wrong choice for some input of size $n$, with probability at most $2^n 2^{-n^2} < 1$ ($2^n$ is the number of possible inputs $x$). That means, there is a at least one choice of random string which is correct for all inputs of a particular size. This string can be given as advice for that particular input size. This proves that BPP $\subset$ P/poly.