CSE 632: Analysis of Algorithms II: Randomized Algorithms **Spring 2024**

Lecture 11: Polynomial Identity Testing

Lecturer: Zongchen Chen

1 Polynomial Identity Testing

We consider the polynomial identity testing problem: Given two polynomials Q and R of degree at most d in n variables $x_1, \dots, x_n$, determine whether $Q = R$ or not.

Note that Q or R might have exponentially many terms. So we assume that we have oracle access to both Q and R; namely, there is an oracle that given values $x_1, \dots, x_n$ it returns $Q(x_1, \dots, x_n)$ and $R(x_1, \dots, x_n)$. In many applications we cannot "write down" the polynomials Q and R completely or explicitly, but we are able to evaluate them efficiently at any given point, and hence such oracles are available.

Consider the polynomial $P = Q - R$, which also has degree at most d, and we can evaluate P at a given point by evaluating both Q and R. Further, $Q = R$ if and only if $P = 0$. We thus obtain the following simpler but equivalent version of polynomial identity testing.

Polynomial identity testing problem: Given oracle access to a polynomial P of degree at most d in n variables $x_1, \dots, x_n$, determine whether $P = 0$ or not.

We give a randomized algorithm for polynomial identity testing.

Algorithm 1 Randomized algorithm for polynomial identity testing

```
1: Choose x_1, ..., x_n independently and u.a.r. from S = {1, 2, ..., 2d}
2: if P(x_1, ..., x_n) = 0 then
       return Yes                                  ▷ Corresponding to P = 0
3: else
       return No                                   ▷ Corresponding to P ≠ 0
4: end if
```

Lemma 1 (Schwartz-Zippel Lemma)**.** *For any finite set S, if $P \neq 0$, then*

$$\Pr\left(P(x_1, \dots, x_n) = 0\right) \leq \frac{d}{|S|}.$$

Proof. Induct on n. For the base case, i.e., $n = 1$, the polynomial $P = P(x_1)$ is univariate. Since the degree of P is at most d, it has at most d (distinct real) roots by the Fundamental Theorem of Algebra. Thus, $\Pr(P(x_1) = 0) \leq \frac{d}{|S|}$.

Suppose the lemma holds for polynomials in $n-1$ variables. We write $P = P(x_1, \dots, x_n)$ as a univariate polynomial in x_1:

$$P(x_1, \dots, x_n) = \sum_{i=0}^{k} A_i(x_2, \dots, x_n) x_1^i,$$

where k is the maximum degree of x_1 in P, and $\deg(A_i) \leq d - i$ for each i. For simplicity, we write $x = (x_1, \dots, x_n)$ and $x_{-1} = (x_2, \dots, x_n)$. By the law of total probability, we deduce that

$$\begin{aligned}\Pr\left(P(x) = 0\right) &= \Pr\left(A_k(x_{-1}) = 0\right) \Pr\left(P(x) = 0 \mid A_k(x_{-1}) = 0\right) + \Pr\left(A_k(x_{-1}) \neq 0\right) \Pr\left(P(x) = 0 \mid A_k(x_{-1}) \neq 0\right) \\ &\leq \Pr\left(A_k(x_{-1}) = 0\right) + \Pr\left(P(x) = 0 \mid A_k(x_{-1}) \neq 0\right). \qquad (1)\end{aligned}$$

By the induction hypothesis, we have

$$\Pr\left(A_k(x_{-1}) = 0\right) \le \frac{d-k}{|S|} \tag{2}$$

since $A_k(x_{-1}) = A_k(x_2, \dots, x_n)$ is a polynomial of degree at most $d-k$ in $n-1$ variables. Meanwhile, for any fixed $x_2, \dots, x_n$ such that $A_k(x_{-1}) \neq 0$, we get a polynomial in x_1:

$$P_1(x_1) = \sum_{i=0}^{k} a_i x_1^i,$$

where $a_i = A_i(x_{-1})$. Since $a_k = A_k(x_{-1}) \neq 0$, $P_1 \neq 0$ is a univariate polynomial of degree k. Again by the Fundamental Theorem of Algebra, P_1 has at most k roots and hence $\Pr(P_1(x_1) = 0 \mid x_{-1}) \le \frac{k}{|S|}$. This holds for all $x_2, \dots, x_n$ satisfying $A_k(x_{-1}) \neq 0$, and therefore we obtain

$$\Pr\left(P(x) = 0 \mid A_k(x_{-1}) \neq 0\right) \le \frac{k}{|S|}. \tag{3}$$

Combining Eqs. (1) to (3), we get $\Pr\left(P(x) = 0\right) \le \frac{d}{|S|}$ as wanted. □

Since for our choice $|S| = 2d$, by the Schwartz-Zippel lemma, we obtain the success and failure probabilities of Algorithm 1 as summarized in the table below.

	Pr(**return** Yes)	Pr(**return** No)
$P = 0$	1	0
$P \neq 0$	$\le \frac{1}{2}$	$\ge \frac{1}{2}$

2 Matrix Multiplication Testing

Recall the matrix multiplication testing problem: Given three matrices $A, B, C \in \mathbb{Z}^{n \times n}$, determine whether $AB = C$ or not.

Consider n polynomials of degree at most 1 (i.e., linear functions) in n variables $x_1, \dots, x_n$ given by

$$P(x_1, \dots, x_n) = \begin{pmatrix} P_1(x_1, \dots, x_n) \\ \vdots \\ P_n(x_1, \dots, x_n) \end{pmatrix} = (AB - C) \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}.$$

Each polynomial P_i has degree at most 1. Furthermore, given values $x_1, \dots, x_n$ we can evaluate $P(x_1, \dots, x_n)$ in $O(n^2)$ time since $(AB - C)x = A(Bx) - Cx$. Observe that $AB = C$ if and only if $P = 0$, i.e., $P_1 = \cdots = P_n = 0$. Therefore, our randomized algorithm for polynomial identity testing recovers Freivalds' algorithm for matrix multiplication testing.

3 Bipartite Perfect Matching

Given a bipartite graph $G = (U \cup V, E)$ where $|U| = |V| = n$, we want to determine whether or not G has a perfect matching.

Definition 2 (Tutte Matrix)**.** For a bipartite graph $G = (U \cup V, E)$ where $U = \{u_1, \dots, u_n\}$ and $V = \{v_1, \dots, v_n\}$, the Tutte matrix of G is an $n \times n$ matrix $A_G = (a_{ij})_{i,j=1}^n$ with entries given by

$$a_{ij} = \begin{cases} x_e, & \text{if } e = u_i v_j \in E \\ 0, & \text{otherwise} \end{cases}$$

where the x_e's are indeterminates (variables).

Example 3. For a bipartite graph $G = (U \cup V, E)$ on 4 vertices where $U = \{u_1, u_2\}$, $V = \{v_1, v_2\}$ and $E = \{u_1v_1, u_2v_1, u_2v_2\}$, the Tutte matrix of G is

$$A_G = \begin{pmatrix} x_{11} & 0 \\ x_{21} & x_{22} \end{pmatrix}$$

where we write $x_{ij} = x_{u_iv_j}$ for simplicity.

The Tutte matrix A_G of a balanced bipartite graph G is a polynomial matrix, meaning that every entry of A_G is a polynomial in x_e's, $e \in E$. The determinant of A_G is given by

$$\det(A_G) = \sum_{\sigma \in S_n} \operatorname{sgn}(\sigma) \prod_{i=1}^{n} a_{i,\sigma(i)},$$

where S_n is the set of $n!$ permutations of $\{1, \ldots, n\}$ and $\operatorname{sgn}(\sigma) \in \{\pm 1\}$ is the sign of a permutation σ given by

$$\begin{aligned} \operatorname{sgn}(\sigma) &= (-1)^{\#\text{ of inversions in } \sigma} \\ &= (-1)^{\#\text{ of even cycles in } \sigma} \\ &= (-1)^{n-(\#\text{ of cycles in } \sigma)}. \end{aligned}$$

Observe that $\det(A_G)$ is a polynomial of degree at most n in m variables $\{x_e : e \in E\}$.

Lemma 4. *A balanced bipartite graph G contains a perfect matching if and only if* $\det(A_G) \neq 0$.

Proof. For every $\sigma \in S_n$, consider two mutually exclusive cases.

(1) If $u_iv_{\sigma(i)} \in E$ for all i, then $M = \{u_1v_{\sigma(1)}, \ldots, u_nv_{\sigma(n)}\}$ forms a perfect matching of G, and we have

$$\prod_{i=1}^{n} a_{i,\sigma(i)} = \prod_{e \in M} x_e.$$

(2) Otherwise, $u_iv_{\sigma(i)} \notin E$ for some i, hence $a_{i,\sigma(i)} = 0$, and we have

$$\prod_{i=1}^{n} a_{i,\sigma(i)} = 0.$$

Let $\mathcal{PM}$ denote the set of all perfect matchings of G. For every perfect matching $M \in \mathcal{PM}$, let $\sigma_M \in S_n$ be the permutation it corresponds to, i.e., $M = \{u_1v_{\sigma(1)}, \ldots, u_nv_{\sigma(n)}\}$. It follows that

$$\begin{aligned} \det(A_G) &= \sum_{\sigma \in S_n} \operatorname{sgn}(\sigma) \prod_{i=1}^{n} a_{i,\sigma(i)} \\ &= \sum_{M \in \mathcal{PM}} \operatorname{sgn}(\sigma_M) \prod_{e \in M} x_e. \end{aligned}$$

Therefore, $\det(A_G) \neq 0$ if and only if $\mathcal{PM} \neq \emptyset$. $\square$

We can evaluate $\det(A_G)$ in $O(n^\omega)$ time where $\omega \approx 2.37$ (currently) is the matrix multiplication exponent. Therefore, by Lemma 4 we can use Algorithm 1 to determine if G has a perfect matching or not.