Lecture 3: Polynomial Identity Testing and Application to Matching

1 Polynomial Identity Testing (PIT)

Definition 1 (Field)**.** *A field $\mathbb{F}$ is a set with two operations, addition ($+$) and multiplication ($\cdot$), satisfying:*

- *Commutative addition and multiplication.*
- *$(\mathbb{F}, +)$ is an abelian group.*
- *$(\mathbb{F} \setminus \{0\}, \cdot)$ is a commutative group.*
- *The distributive law holds: $a(b + c) = ab + ac$.*

Examples: Reals ($\mathbb{R}$), complex numbers ($\mathbb{C}$), rationals ($\mathbb{Q}$).

Definition 2 (Finite Fields)**.** *A finite field is a field with a finite number of elements.*

- *$\mathbb{Z}_p = \{0, 1, 2, \ldots, p-1\}$ with addition and multiplication modulo a prime p.*
- *Every finite field has size p^k for some prime p and integer $k \geq 1$.*

1.1 The PIT Problem

Problem: Given a multivariate polynomial $P(x_1, x_2, \ldots, x_n)$ over a field $\mathbb{F}$ via a "black box", we want to know if P is identically zero, i.e., $P \equiv 0$.

By "black box", we mean we can ask for values of $P(a_1, a_2, \ldots, a_n)$ for any given field elements $a_1, \ldots, a_n \in \mathbb{F}$. More formally, we are given a circuit that evaluates P.

Example: $P(x_1, x_2, x_3) = 2(x_1 - x_2)x_3 + 2x_3x_2 + 2x_1^2 - (x_1 + x_3)^2$.

Checking if two polynomials P_1 and P_2 are equal is the same as checking if $P_1 - P_2 \equiv 0$.

Definition 3 (Multivariate Polynomial)**.** *A multivariate polynomial is a sum of monomials with coefficients. A monomial is a product of powers of the variables:*

$$x_1^{i_1} x_2^{i_2} \cdots x_n^{i_n}$$

where $i_1, i_2, \ldots, i_n \geq 0$ are integers. The degree of a monomial is $\sum_{j=1}^{n} i_j$. The degree of a polynomial P, denoted $\deg(P)$*, is the maximum degree among its monomials.*

Example: *For $P(x_1, x_2) = x_1^3 x_2^4 + x_1^{10} + x_1^9 x_2^3$,* $\deg(P) = 12$.

1.2 Algorithms for PIT

1.2.1 Univariate Case

A polynomial with only one variable, x.

Theorem 1. *A degree d non-zero univariate polynomial $P(x)$ over a field $\mathbb{F}$ has at most d roots.*

Deterministic Algorithm:

1. Pick any set of $d+1$ distinct field elements $a_1, a_2, \ldots, a_{d+1}$.
2. Evaluate $P(a_1), \ldots, P(a_{d+1})$.
3. If all are zero, then $P \equiv 0$. Otherwise, $P \not\equiv 0$.

Randomized Algorithm:

1. Let $S \subseteq \mathbb{F}$ be a finite set.
2. Pick $a \in S$ uniformly at random.
3. If $P(a) = 0$, output $P \equiv 0$.
4. Else, output $P \not\equiv 0$.

Analysis:

- If $P \equiv 0$, the algorithm is always correct.
- If $P \not\equiv 0$, the algorithm makes an error only if it picks a root of P.

$$\Pr[\text{Algorithm outputs YES} | P \not\equiv 0] \leq \frac{d}{|S|}$$

By choosing $|S|$ large, we can make the error probability small. We can repeat the test to decrease the error probability exponentially.

The Catch: If working with reals, we cannot "really" pick a fully random real number or evaluate $P(a)$ exactly due to precision issues. A common technique is to work over a finite field. If we know the coefficients are integers with maximum absolute value B, we can choose a prime $p > \max(B, d)$ and perform all computations in $\mathbb{Z}_p$. Then $P \equiv 0$ over $\mathbb{Q}$ iff $P \equiv 0$ over $\mathbb{Z}_p$.

1.2.2 Multivariate Case

For multivariate polynomials, there is no easy deterministic algorithm known. But randomization still works!

Lemma 1 (Schwarz-Zippel Lemma)**.** *Let $P(x_1, \ldots, x_n)$ be a non-zero multivariate polynomial of total degree d over a field $\mathbb{F}$. Let $S \subseteq \mathbb{F}$ be a finite set. If we pick $a_1, \ldots, a_n$ uniformly and independently at random from S, then*

$$\Pr[P(a_1, \ldots, a_n) = 0] \leq \frac{d}{|S|}$$

Proof Sketch by Induction on n. **Base Case ($n = 1$):** $P(x_1)$ is a univariate polynomial of degree at most d. It has at most d roots. The probability of picking a root is $\leq d/|S|$.

Inductive Step: Assume the lemma holds for $n-1$ variables. We can write P as a polynomial in x_n:

$$P(x_1, \ldots, x_n) = \sum_{j=0}^{k} Q_j(x_1, \ldots, x_{n-1}) x_n^j$$

where k is the highest power of x_n and $Q_k \not\equiv 0$. Let $\deg(Q_k) \leq d - k$. Let A be the event $P(a_1, \ldots, a_n) = 0$ and B be the event $Q_k(a_1, \ldots, a_{n-1}) = 0$.

$$\begin{aligned} \Pr[A] &= \Pr[A|B] \Pr[B] + \Pr[A|\neg B] \Pr[\neg B] \\ &\leq 1 \cdot \Pr[B] + \Pr[A|\neg B] \cdot 1 \end{aligned}$$

By the inductive hypothesis, $\Pr[B] \leq \frac{\deg(Q_k)}{|S|} \leq \frac{d-k}{|S|}$. If $\neg B$ occurs, then $P(a_1, \ldots, a_{n-1}, x_n)$ is a non-zero univariate polynomial in x_n of degree at most k. The probability that a_n is a root is $\leq k/|S|$. So, $\Pr[A|\neg B] \leq \frac{k}{|S|}$.

$$\Pr[A] \leq \frac{d-k}{|S|} + \frac{k}{|S|} = \frac{d}{|S|}$$

□

Major Open Question: Is there a deterministic polynomial-time algorithm for PIT? This is a central question in derandomization. A positive answer would imply major results in computational complexity, such as circuit lower bounds.

2 Application to Matchings

Definition 4 (Matching)**.** *Given a graph $G = (V, E)$, a matching is a subset of edges $M \subseteq E$ such that no two edges in M share a vertex.*

- *A matching is* ***perfect*** *if every vertex is incident to exactly one edge in M. This implies $|M| = |V|/2$.*

Matching is a fundamental problem with many applications. Algorithmic questions include finding a perfect matching, a maximum cardinality matching, or a maximum weight matching. For bipartite graphs, these problems can be solved using network flow algorithms. For general graphs, the algorithms are more complex (e.g., using "blossoms"). We will see an algebraic randomized approach.

2.1 Bipartite Perfect Matching

Let $G = (U \cup V, E)$ be a bipartite graph with $|U| = |V| = n$. We want to determine if G has a perfect matching. Let $U = \{u_1, \ldots, u_n\}$ and $V = \{v_1, \ldots, v_n\}$.

Define the $n \times n$ **Edmonds matrix** A:

$$A_{ij} = \begin{cases} x_{ij} & \text{if } (u_i, v_j) \in E \\ 0 & \text{otherwise} \end{cases}$$

where each x_{ij} is a distinct variable.

The determinant of A, $\det(A)$, is a multivariate polynomial.

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

where S_n is the set of all permutations of $\{1, \ldots, n\}$. Each term in this sum corresponds to a permutation σ. The product $\prod_{i=1}^{n} A_{i,\sigma(i)}$ is non-zero if and only if for all i, the edge $(u_i, v_{\sigma(i)})$ exists in G. The set of edges $\{(u_i, v_{\sigma(i)}) | i = 1, \ldots, n\}$ is exactly a perfect matching.

Lemma 2. *The graph G has a perfect matching if and only if* $\det(A) \not\equiv 0$.

Proof. If G has no perfect matching, then for any permutation σ, there is at least one i such that $(u_i, v_{\sigma(i)}) \notin E$. Thus $A_{i,\sigma(i)} = 0$, and the corresponding term in the determinant is zero. So, $\det(A) \equiv 0$.

If G has a perfect matching $M = \{(u_i, v_{\sigma(i)}) | i = 1, \ldots, n\}$ for some permutation σ, then the term $\prod_{i=1}^{n} A_{i,\sigma(i)} = \prod_{i=1}^{n} x_{i,\sigma(i)}$ is a non-zero monomial in $\det(A)$. Since each permutation corresponds to a unique set of variables, this monomial cannot be cancelled by any other term. Thus, $\det(A) \not\equiv 0$. □

This reduces the perfect matching problem to PIT. **Randomized Algorithm for Bipartite Perfect Matching:**

1. Construct the Edmonds matrix A with variables x_{ij}.
2. The degree of $\det(A)$ is n.
3. Choose a field $\mathbb{F}$ with $|\mathbb{F}| > 2n$, for example $\mathbb{Z}_p$ with a prime $p > 2n$.
4. For each variable x_{ij}, pick a value a_{ij} uniformly at random from $\mathbb{F}$.
5. Compute the determinant of the resulting numerical matrix.

6. If the determinant is non-zero, output YES (a perfect matching exists).

7. If the determinant is zero, output NO.

By the Schwarz-Zippel Lemma, the probability of error (saying NO when there is a matching) is at most $\frac{n}{|\mathbb{F}|} < \frac{n}{2n} = \frac{1}{2}$. The determinant can be computed efficiently, e.g., in $O(n^\omega)$ time where $\omega < 2.38$ is the matrix multiplication exponent. This approach is highly parallelizable.

2.2 General Graph Perfect Matching

Let $G = (V, E)$ be a general graph with $n = |V|$ vertices (assume n is even). Define the $n \times n$ skew-symmetric **Tutte matrix** T:

$$T_{ij} = \begin{cases} x_{ij} & \text{if } \{i,j\} \in E \text{ and } i < j \\ -x_{ji} & \text{if } \{i,j\} \in E \text{ and } i > j \\ 0 & \text{otherwise} \end{cases}$$

Here, x_{ij} and x_{ji} are the same variable, so $T_{ij} = -T_{ji}$.

Theorem 2 (Tutte, 1947)**.** *A graph G has a perfect matching if and only if* $\det(T) \not\equiv 0$.

Proof Sketch. The proof is more involved because cancellations are essential. A term $\operatorname{sgn}(\sigma) \prod_{i=1}^{n} T_{i,\sigma(i)}$ in $\det(T)$ is non-zero only if σ is a permutation with no fixed points (since $T_{ii} = 0$) and $\{i, \sigma(i)\} \in E$ for all i. Such a permutation decomposes into a set of disjoint cycles in the graph.

- If σ contains an odd-length cycle, its term in the determinant expansion gets cancelled. Consider the permutation σ' obtained by reversing an odd cycle in σ. One can show that $\operatorname{sgn}(\sigma) \prod T_{i,\sigma(i)} = -\operatorname{sgn}(\sigma') \prod T_{i,\sigma'(i)}$, and these two terms cancel out.

- Therefore, all terms corresponding to permutations with odd cycles sum to zero. The determinant is the sum over permutations σ whose cycle decomposition consists only of even-length cycles.

- If G has a perfect matching $M = \{\{i_1, j_1\}, \ldots, \{i_{n/2}, j_{n/2}\}\}$, this corresponds to a permutation σ consisting of $n/2$ cycles of length 2 (transpositions). All cycles are even. The corresponding term is non-zero. It can be shown that these terms for permutations with only even cycles do not all cancel out.

- Conversely, if $\det(T) \not\equiv 0$, there must be a non-cancelling term, which must correspond to a permutation σ with only even-length cycles. A set of edges corresponding to such a σ (a 2-factor with only even cycles) can be shown to contain a perfect matching (since any even cycle can be decomposed into two perfect matchings on its vertices).

□

The same randomized algorithm using PIT for $\det(T)$ can be used to decide if a general graph has a perfect matching. This remains one of the most efficient parallel algorithms for the problem.

2.3 Finding a Matching and the Isolation Lemma

The PIT-based algorithms are decision algorithms (they say if a matching exists). To find a matching (the search problem), a standard reduction from search to decision is sequential. For a parallel algorithm, a different idea is needed.

The **Isolation Lemma** (Mulmuley, Vazirani, Vazirani) is a powerful tool for this.

Lemma 3 (Isolation Lemma)**.** *Let $U = \{1, 2, \ldots, m\}$ be a ground set, and let $\mathcal{F} \subseteq 2^U$ be a non-empty family of subsets. Assign a weight w_i to each element $i \in U$, chosen independently and uniformly at random from $\{1, 2, \ldots, 2m\}$. For a set $S \subseteq U$, its weight is $W(S) = \sum_{i \in S} w_i$. Then,*

$$\Pr[\textit{There is a unique minimum weight set in } \mathcal{F}] \geq \frac{1}{2}$$

By assigning random weights to the edges of the graph, the isolation lemma guarantees that with good probability, there will be a unique perfect matching of minimum weight. This uniqueness can be exploited in a parallel algorithm to find that matching.