10.1 Polynomial identity testing

10.1.1 Matrix multiplication

We want to check matrix multiplication. We have \( n \times n \) matrices \( A, B \) and \( C \), and we want to check if \( A \times B = C \).

**Naive approach:** Compute \( A \times B \) in time of \( O(n^3) \) (matrix multiplication)

**Randomized approach:** Choose a random vector \( r = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_n \end{bmatrix} \) where each \( r_i \) is independently and uniformly at random from \( S = \{1, 2, \ldots k\} \).

Compute \( (AB)r \) and \( Cr \) and check if both are equal. Time complexity for this method = \( O(n^2) \)

**Claim:** \( \Pr_r((AB)r = Cr | AB \neq C) \leq 1/k \)

If we run \( t \) trials, we can boost this probability to \( k^{-t} \)

**Proof:** Assume \( AB \neq C \). So \( D = AB - C \neq 0 \).

Assume \( d_{11} \neq 0 \) (if it’s not, we can relabel rows and columns to make it true)

\[
\Pr(D_r) = 0 \leq \Pr((D_r)_1 = 0) \leq \Pr(r_1 = S^*) \leq \frac{1}{k}
\]

\[\vdots\]

\[ (Dr)_1 = \sum_{i=1}^{n} d_{i1} r_i = 0 \]

\[ \Rightarrow r_1 = \frac{-1}{d_{11}}(d_{12} r_2 + d_{13} r_3 + \cdots + d_{1n} r_n) = S^* \]

10.1.2 Polynomial Equality Testing

Now, let’s consider two polynomials \( P \) & \( Q \) over \( n \) variables \( X_1, \ldots, X_n \). We want to know if \( P = Q \).

We assume “oracle” access to \( P \) and \( Q \), i.e., for a given \( X = X_1, \ldots, X_n \), we can evaluate \( P \) and \( Q \) at \( X \) efficiently.

**Proof:** Assume \( R \neq 0 \). Induct on \( n \)

Base case: \( n = 1 \), \( R(x_1) \) univariate polynomial of degree \( \leq d \) \( \implies \leq d \) roots.

General: Take \( x_1 \) and term of max degree in \( x \), say \( j \). Factor out \( x_1^j \)

\[
R(x_1, \ldots, x_n) = x_1^j \underbrace{(M(x_2, \ldots, x_n)) + N(x_1, \ldots, x_n)}_{\text{n-1 variables max. deg. of } x_i < j}
\]
Algorithm 1: Schwarz-Zippel algorithm

1 Consider \( R = P - Q \). Check if \( R = 0 \)?
2 Choose \( x_i \) uniformly at random from \( S = \{1, \cdots, k\} \);
3 if \( R(x_1, \cdots, x_n) = 0 \) then
4 output YES;
5 else
6 output NO;
7 \( \Pr(R(x_1, \cdots, x_n) = 0 | R \neq 0) \leq \frac{d}{k} \) (\( d = \# \text{ of roots} \));
8 if \( k \geq 2d \) then
9 False positive probability \( \leq \frac{1}{2} \);
10 and with \( t \) trials \( \Rightarrow 2^{-t} \);

Using Principle of Deferred Decisions, fix \( x_2, \cdots, x_n \) and consider \( x_1 \).
Let event \( \xi \) be \( M(x_2, \cdots, x_n) = 0 \)
Now:
\[
\Pr(R(x_1, \cdots, x_n) = 0) = \Pr(R(x_1, \cdots, x_n) = 0 | \xi) \Pr(\xi) + \Pr(R(x_1, \cdots, x_n) = 0 | \bar{\xi}) \Pr(\bar{\xi})
\]
Taking the bigger value for both, we get:
\[
\Pr(R(x_1, \cdots, x_n) = 0) = \Pr(\xi) + \Pr(R(x_1, \cdots, z_n) | \bar{\xi}) \Pr(\bar{\xi})
\]
Now,
\[
\Pr\xi = \Pr(M(x_2, \cdots, x_n) = 0) \leq \frac{d-j}{k}
\]

where \( d = \) original degree, \( j = \) degree when \( x_1 \) factored out
Using Principle of Deferred Decisions, plug in \( x_2, \cdots, x_n \) in the \( R \) equation. \( R \) remains univariate now with just one unknown \( x_1 \). Thus we can can apply base case here.
\[
\deg(R(x_1)) \leq j \implies \Pr(R(x_1, \cdots, x_n) = 0 | \bar{\xi}) = \Pr(R(x_1) = 0 | x_2, \cdots, x_n, \bar{\xi}) \leq \frac{j}{k}
\]
Using these values in the original equation,
\[
\Pr(R(x_1, \cdots, x_n) = 0) \leq \frac{d-j}{k} + \frac{j}{k} = \frac{d}{k}
\]

Note: It is not necessary to choose from \( \{1, \cdots, k\} \). It is important to choose from \( 'k' \) different numbers.

10.1.3 Perfect Matching

Bipartite graph \( G = (L \cup R, E) \). Does \( G \) have a perfect matching?

For any edge \( (i,j) \) in \( E \): \( M_G = \begin{cases} x_{ij} & \text{if } (i,j) \in E, \\ 0 & \text{otherwise} \end{cases} \).

Claim: \( \det(M) \Leftrightarrow G \) has a perfect matching.

Proof: Test if \( \det(M) \neq 0 \): choose \( x_{ij} \) uniformly at random from \( \{1, \cdots, 2n\} \).
\[ L_\Rightarrow: G \text{ has a perfect matching } P, \text{ every perfect matching } P \text{ has a unique term } \prod_{(i, j) \in P}. \]

\[ \Rightarrow: \det(M) \neq 0 \]

\[ \det(M) = \sum_{\sigma \in S_n} \text{sgn}(\sigma) \prod_{i=1}^{n} M_{i\sigma(i)} \]

where:

\[ S_n = \text{permutations of } \{1, \cdots, n\} \]

\[ \text{sgn}(\sigma) = (-1)^{\\text{nb. of inversions in } \sigma} = (-1)^{\\text{nb. of even cycles in } \sigma} = (-1)^{n-\text{nb. of cycles in } \sigma} \]

Test if \( G \) has a perfect matching?

1. Assume \( G \) has a perfect matching. \( P \) corresponds to a permutation.
   \( \Pi \) gives \( x_{ij} \) (not zero), each edge gives a distinct variable. Every perfect matching has a unique term \( \Pi_{(i,j) \in P}x_{ij} \)
   There has to be at least one non-zero term, thus making \( \det(M) \neq 0 \).

2. Assume \( \det(M) \neq 0 \).
   \( \Pi \) gives non-zero terms for each \((i, j) \in E\). Because all edges of the perfect matching belong to the graph \( G \), all edges of the perfect matching exist and \( \Pi \) gives non-zero values for each of those.
   \( \Rightarrow \) There exists a perfect matching.

**Algorithm 2**: Test if \( G \) has a perfect matching.

1. for each edge \((i, j) \in E\), choose \( x_{ij} \) u.a.r. from \( \{1, \cdots, 2n\} \) do
2. Compute \( \det(M) \): \( \Pr(\det(M) = 0 | G \text{ has a perfect matching}) \leq \frac{1}{2} \).

Run it \( t \) times to boost this probability to \( \leq 2^{-t} \)

\( G = (V, E), \text{ edge } (i, j) \in E. \)

Induced subgraph on \( V \setminus \{i, j\}, M_{ij} = M \) with row \( i \) and column \( j \) removed.

Check if \( \det(M_{ij}) \neq 0? \) (using the algorithm described above)

Recurse on the smaller graph.

Time complexity: \( O(|E|) \) rounds

**Question**: Can it be done in parallel (check all edges at the same time to see which ones belong to the perfect matching)?

**Problem**: Every edge might be in ‘a’ perfect matching, but it does not necessarily mean that they belong to the same one.

**Solution**: We can find a unique perfect matching with minimum weight. Check if \((i, j) \in E\) is in the minimum weight Perfect Matching (check value of the determinant to get the minimum weight P.M.). All determinant evaluations will go towards the same unique perfect matching.
Algorithm 3: Mulmuley, Vazirani, Vazirani, ’87

1. Let $S = \{x_1, \cdots, x_m\}$. Subsets $S_1, \ldots, S_k$ of $S$;
2. Randomly assign $\omega = S \rightarrow \{1, \ldots, l\}$: $\omega(S_i) = \sum_{x \in S_i} \omega(x)$;

Lemma 10.1 (Isolation Lemma) From algorithm 3:

$$\Pr(\text{unique set } S_i \text{ of min. weight}) \geq 1 - \frac{m}{l}$$

where $S_i$s are perfect matchings.

Proof: We say that $X \in S$ is tied if $\min_{X \in S_i} \omega(S_i) = \min_{X \notin S_i} \omega(S_i)$

Unique subset $S_i$ of minimum weight iff no $X$ is tied.

$\Pr(X \text{ is tied}) = \Pr(\omega(x) = \omega^- - \omega^+) = \frac{1}{l}$.

Fix $\omega(y)$ for all $y \in S$ such that $y \neq x$.

$\Pr(\text{not unique subset } S_i \text{ of min weight}) = \Pr(\text{some } X \text{ is tied}) = \sum_{Y \in S} \Pr(Y \text{ is tied}) \leq \frac{m}{l}$.

$\implies \Pr(\text{unique}) \geq 1 - \frac{m}{l}$.