Identities and Complexity

Nitin Saxena

Centrum voor Wiskunde en Informatica
Amsterdam

NVTI Theory Day 2007
Utrecht
Motivation

Outline

Motivation

Identity Testing

Constant Depth Circuits

Conclusion
Motivation

**Identities**

- High School algebra teaches us lots of useful algebraic identities.
- For example,
  \[ x^3 + y^3 + z^3 - 3xyz = (x + y + z)(x^2 + y^2 + z^2 - xy - yz - zx). \]
- Lebesgue identity:
  \[
  (a^2 + b^2 + c^2 + d^2)^2 = (a^2 + b^2 - c^2 - d^2)^2 + (2ac + 2bd)^2 + (2ad - 2bc)^2
  \]
Motivation

Identities

- High School algebra teaches us lots of useful algebraic identities.
- For example,
  \[x^3 + y^3 + z^3 - 3xyz = (x + y + z)(x^2 + y^2 + z^2 - xy - yz - zx)\].
- Lebesgue identity:
  \[
  (a^2 + b^2 + c^2 + d^2)^2 = (a^2 + b^2 - c^2 - d^2)^2 + (2ac + 2bd)^2 + (2ad - 2bc)^2
  \]
Identities

• High School algebra teaches us lots of useful algebraic identities.

• For example,
  \[ x^3 + y^3 + z^3 - 3xyz = (x + y + z)(x^2 + y^2 + z^2 - xy - yz - zx). \]

• Lebesgue identity:

\[
(a^2 + b^2 + c^2 + d^2)^2 = (a^2 + b^2 - c^2 - d^2)^2 + (2ac + 2bd)^2 + (2ad - 2bc)^2
\]
Motivation

Identities

• Identity communicated by Euler in a letter to Goldbach on April 15, 1750:
\[(a_1^2 + a_2^2 + a_3^2 + a_4^2)(b_1^2 + b_2^2 + b_3^2 + b_4^2) = (a_1 b_1 - a_2 b_2 - a_3 b_3 - a_4 b_4)^2 + (a_1 b_2 + a_2 b_1 + a_3 b_4 - a_4 b_3)^2 + (a_1 b_3 - a_2 b_4 + a_3 b_1 + a_4 b_2)^2 + (a_1 b_4 + a_2 b_3 - a_3 b_2 + a_4 b_1)^2\]

• All these can be checked by expansion.
Motivation

Identities

- Identity communicated by Euler in a letter to Goldbach on April 15, 1750:
  \[
  (a_1^2 + a_2^2 + a_3^2 + a_4^2)(b_1^2 + b_2^2 + b_3^2 + b_4^2) = \\
  (a_1 b_1 - a_2 b_2 - a_3 b_3 - a_4 b_4)^2 + (a_1 b_2 + a_2 b_1 + a_3 b_4 - a_4 b_3)^2 + \\
  (a_1 b_3 - a_2 b_4 + a_3 b_1 + a_4 b_2)^2 + (a_1 b_4 + a_2 b_3 - a_3 b_2 + a_4 b_1)^2
  \]

- All these can be checked by expansion.
Bigger Identities

• Let $p$ be an odd prime number. Then:

$$\sum_{i=1}^{p} \prod_{a_1, \ldots, a_m \in \mathbb{F}_p} (y + a_1x_1 + \cdots + a_mx_m) = 0$$

• The polynomial on the LHS has degree: $p^{m-1}$.

• A naive expansion of the above produces exponentially many terms.

• Then how do we check the above identity efficiently?
**Motivation**

**Bigger Identities**

- Let $p$ be an odd prime number. Then:

$$\sum_{i=1}^{p} \prod_{a_1,\ldots,a_m \in \mathbb{F}_p \atop a_1 + \cdots + a_m = i \pmod{p}} (y + a_1 x_1 + \cdots + a_m x_m) = 0$$

- The polynomial on the LHS has degree: $p^{m-1}$.
- A naive expansion of the above produces exponentially many terms.
- Then how do we check the above identity efficiently?
Bigger Identities

• Let $p$ be an odd prime number. Then:

$$
\sum_{i=1}^{p} \prod_{a_1,\ldots,a_m \in \mathbb{F}_p \atop a_1 + \cdots + a_m = i \pmod{p}} (y + a_1 x_1 + \cdots + a_m x_m) = 0
$$

• The polynomial on the LHS has degree: $p^{m-1}$.
• A naive expansion of the above produces exponentially many terms.
• Then how do we check the above identity efficiently?
Bigger Identities

- Let $p$ be an odd prime number. Then:
  \[ \sum_{i=1}^{p} \prod_{a_1, \ldots, a_m \in \mathbb{F}_p} (y + a_1 x_1 + \cdots + a_m x_m) = 0 \]
  \[ a_1 + \cdots + a_m = i \pmod{p} \]

- The polynomial on the LHS has degree: $p^{m-1}$.
- A naive expansion of the above produces exponentially many terms.
- Then how do we check the above identity efficiently?
**Randomness Helps**

- Evaluate the above polynomial at a *random* point.
- It can be shown that “with high probability” the polynomial evaluates to zero iff it is an identity!
- But can this identity testing be done efficiently without using randomness?
Randomness Helps

- Evaluate the above polynomial at a random point.
- It can be shown that “with high probability” the polynomial evaluates to zero iff it is an identity!
- But can this identity testing be done efficiently without using randomness?
Randomness Helps

- Evaluate the above polynomial at a *random* point.
- It can be shown that “with high probability” the polynomial evaluates to zero iff it is an identity!
- But can this identity testing be done efficiently without using randomness?
Randomness Helps ??

- There are many problems with nice randomized efficient algorithms.
- Like, Identity testing, Primality testing, Polynomial factorization, Quicksort, Min-cut,......
- But there is a belief that randomness in polynomial time algorithms is always dispensable. In short: “God does not play dice...."
• There are many problems with nice randomized efficient algorithms.
• Like, Identity testing, Primality testing, Polynomial factorization, Quicksort, Min-cut,......
• But there is a belief that randomness in polynomial time algorithms is always dispensable. In short:
  “God does not play dice...."
There are many problems with nice randomized efficient algorithms.

Like, Identity testing, Primality testing, Polynomial factorization, Quicksort, Min-cut,......

But there is a belief that randomness in polynomial time algorithms is *always* dispensable. In short:

"God does not play dice...."
Motivation

Randomness Helps ??

- There are many problems with nice randomized efficient algorithms.
- Like, Identity testing, Primality testing, Polynomial factorization, Quicksort, Min-cut,......
- But there is a belief that randomness in polynomial time algorithms is always dispensable. In short:

  “God does not play dice....”
Motivation

Randomness Helps ??

- Impagliazzo-Wigderson ’96 showed that if there are “hard” functions in E then polynomial time randomized algorithms can be derandomized.
- Primality testing was successfully derandomized by Agrawal-Kayal-S in 2002.
- After Primality testing, arguably, the next most important problem waiting to be derandomized is identity testing.
Motivation

Randomness Helps ??

- Impagliazzo-Wigderson ’96 showed that if there are “hard” functions in E then polynomial time randomized algorithms can be derandomized.

- Primality testing was successfully derandomized by Agrawal-Kayal-S in 2002.

- After Primality testing, arguably, the next most important problem waiting to be derandomized is identity testing.
Randomness Helps ??

- Impagliazzo-Wigderson '96 showed that if there are “hard” functions in E then polynomial time randomized algorithms can be derandomized.
- Primality testing was successfully derandomized by Agrawal-Kayal-S in 2002.
- After Primality testing, arguably, the next most important problem waiting to be derandomized is identity testing.
OUTLINE

Motivation

Identity Testing

Constant Depth Circuits

Conclusion
FORMALIZING IDENTITY TESTING

- We can assume that our polynomial expression is given in the form of an **Arithmetic circuit** $C$:

- **Identity testing** is the problem of checking whether a given circuit is zero or not.
We can assume that our polynomial expression is given in the form of an Arithmetic circuit $C$:

Identity testing is the problem of checking whether a given circuit is zero or not.
Identity Testing

Formalizing Identity Testing

- We can assume that our polynomial expression is given in the form of an Arithmetic circuit $C$:

- Identity testing is the problem of checking whether a given circuit is zero or not.
A Randomized Solution

- Schwartz ’80, Zippel ’79 gave a randomized algorithm for identity testing.

- Given an arithmetic circuit \( C(x_1, \ldots, x_n) \in \mathbb{F}[x_1, \ldots, x_n] \):
  - Pick a random tuple \((\alpha_1, \ldots, \alpha_n) \in \mathbb{F}^n\).
  - Return YES iff \( C(\alpha_1, \ldots, \alpha_n) = 0 \).

- Clearly, this can be done in time polynomial in the size of \( C \).
A Randomized Solution

- **Schwartz ’80, Zippel ’79** gave a randomized algorithm for identity testing.
- Given an arithmetic circuit $C(x_1, \ldots, x_n) \in \mathbb{F}[x_1, \ldots, x_n]$:
  - Pick a random tuple $(\alpha_1, \ldots, \alpha_n) \in \mathbb{F}^n$.
  - Return YES iff $C(\alpha_1, \ldots, \alpha_n) = 0$.
- Clearly, this can be done in time polynomial in the size of $C$. 

A Randomized Solution

- **Schwartz ’80, Zippel ’79** gave a randomized algorithm for identity testing.
- Given an arithmetic circuit $C(x_1, \ldots, x_n) \in \mathbb{F}[x_1, \ldots, x_n]$:
  - Pick a random tuple $(\alpha_1, \ldots, \alpha_n) \in \mathbb{F}^n$.
  - Return YES iff $C(\alpha_1, \ldots, \alpha_n) = 0$.
- Clearly, this can be done in time polynomial in the size of $C$. 
A Randomized Solution

• Schwartz ’80, Zippel ’79 gave a randomized algorithm for identity testing.

• Given an arithmetic circuit $C(x_1, \ldots, x_n) \in \mathbb{F}[x_1, \ldots, x_n]$:
  • Pick a random tuple $(\alpha_1, \ldots, \alpha_n) \in \mathbb{F}^n$.
  • Return YES iff $C(\alpha_1, \ldots, \alpha_n) = 0$.

• Clearly, this can be done in time polynomial in the size of $C$. 
A Randomized Solution

- Schwartz ’80, Zippel ’79 gave a randomized algorithm for identity testing.
- Given an arithmetic circuit $C(x_1, \ldots, x_n) \in \mathbb{F}[x_1, \ldots, x_n]$:
  - Pick a random tuple $(\alpha_1, \ldots, \alpha_n) \in \mathbb{F}^n$.
  - Return YES iff $C(\alpha_1, \ldots, \alpha_n) = 0$.
- Clearly, this can be done in time polynomial in the size of $C$. 
A Randomized Solution

Proof of Correctness:

- If $C$ is a zero circuit then clearly the algorithm returns YES for any choice of $(\alpha_1, \ldots, \alpha_n) \in \mathbb{F}^n$.
- Say, $C(x_1, \ldots, x_n)$ is computing a nonzero polynomial of total degree $d$.
- It can be shown that:

$$\Pr_{(\alpha_1, \ldots, \alpha_n) \in \mathbb{F}^n} [C(\alpha_1, \ldots, \alpha_n) = 0] \leq \frac{d}{\#\mathbb{F}}$$

- Thus, for a suitably large $\mathbb{F}$, $\frac{d}{\#\mathbb{F}} \leq \frac{1}{2}$.
- Thus, with a good chance we will pick a non-root of $C$. 
A Randomized Solution

Proof of Correctness:

- If $C$ is a zero circuit then clearly the algorithm returns YES for any choice of $(\alpha_1, \ldots, \alpha_n) \in \mathbb{F}^n$.
- Say, $C(x_1, \ldots, x_n)$ is computing a nonzero polynomial of total degree $d$.
- It can be shown that:

$$\text{Prob}_{(\alpha_1, \ldots, \alpha_n) \in \mathbb{F}^n} [C(\alpha_1, \ldots, \alpha_n) = 0] \leq \frac{d}{\#\mathbb{F}}$$

- Thus, for a suitably large $\mathbb{F}$, $\frac{d}{\#\mathbb{F}} \leq \frac{1}{2}$.
- Thus, with a good chance we will pick a non-root of $C$. 
A Randomized Solution

Proof of Correctness:

- If $C$ is a zero circuit then clearly the algorithm returns YES for any choice of $(\alpha_1, \ldots, \alpha_n) \in \mathbb{F}^n$.
- Say, $C(x_1, \ldots, x_n)$ is computing a nonzero polynomial of total degree $d$.
- It can be shown that:
  \[
  \text{Prob}_{(\alpha_1, \ldots, \alpha_n) \in \mathbb{F}^n} [C(\alpha_1, \ldots, \alpha_n) = 0] \leq \frac{d}{\#\mathbb{F}}
  \]
- Thus, for a suitably large $\mathbb{F}$, $\frac{d}{\#\mathbb{F}} \leq \frac{1}{2}$.
- Thus, with a good chance we will pick a non-root of $C$. 
A Randomized Solution

Proof of Correctness:

• If $C$ is a zero circuit then clearly the algorithm returns YES for any choice of $(\alpha_1, \ldots, \alpha_n) \in \mathbb{F}^n$.

• Say, $C(x_1, \ldots, x_n)$ is computing a nonzero polynomial of total degree $d$.

• It can be shown that:

$$\text{Prob}_{(\alpha_1, \ldots, \alpha_n) \in \mathbb{F}^n} [C(\alpha_1, \ldots, \alpha_n) = 0] \leq \frac{d}{\#\mathbb{F}}$$

• Thus, for a suitably large $\mathbb{F}$, $\frac{d}{\#\mathbb{F}} \leq \frac{1}{2}$.

• Thus, with a good chance we will pick a non-root of $C$. 
A Randomized Solution

Proof of Correctness:

• If $C$ is a zero circuit then clearly the algorithm returns YES for any choice of $(\alpha_1, \ldots, \alpha_n) \in \mathbb{F}^n$.

• Say, $C(x_1, \ldots, x_n)$ is computing a nonzero polynomial of total degree $d$.

• It can be shown that:

  $$\text{Prob}((\alpha_1, \ldots, \alpha_n) \in \mathbb{F}^n \mid C(\alpha_1, \ldots, \alpha_n) = 0) \leq \frac{d}{\#\mathbb{F}}$$

• Thus, for a suitably large $\mathbb{F}$, $\frac{d}{\#\mathbb{F}} \leq \frac{1}{2}$.

• Thus, with a good chance we will pick a non-root of $C$. 
Big question here: **Can we do identity testing in deterministic polynomial time?**
Identity testing is instrumental in many complexity theory results:

- Graph matching problems have efficient randomized parallel algorithms (Lovasz ’79).
- PSPACE has interactive protocols (Shamir ’90).
- NEXP has two-prover interactive protocols (Babai-Fortnow-Lund ’90).
- The first deterministic polynomial time Primality test was based on checking whether \((x + 1)^n - (x^n + 1) = 0 \pmod{n}\) (Agrawal-Kayal-S ’02).
Identity testing is instrumental in many complexity theory results:

- Graph matching problems have efficient randomized parallel algorithms (Lovasz ’79).
- PSPACE has interactive protocols (Shamir ’90).
- NEXP has two-prover interactive protocols (Babai-Fortnow-Lund ’90).
- The first deterministic polynomial time Primality test was based on checking whether $(x + 1)^n - (x^n + 1) = 0 \pmod{n}$ (Agrawal-Kayal-S ’02).
Identity testing is instrumental in many complexity theory results:

- Graph matching problems have efficient randomized parallel algorithms (Lovasz '79).
- PSPACE has interactive protocols (Shamir '90).
- NEXP has two-prover interactive protocols (Babai-Fortnow-Lund '90).
- The first deterministic polynomial time Primality test was based on checking whether \((x + 1)^n - (x^n + 1) = 0 \pmod{n}\) (Agrawal-Kayal-S '02).
Connections

Identity testing is instrumental in many complexity theory results:

- Graph matching problems have efficient randomized parallel algorithms (Lovasz ’79).
- PSPACE has interactive protocols (Shamir ’90).
- NEXP has two-prover interactive protocols (Babai-Fortnow-Lund ’90).
- The first deterministic polynomial time Primality test was based on checking whether \((x + 1)^n - (x^n + 1) = 0 \pmod{n}\) (Agrawal-Kayal-S ’02).
Deeper Connections

- (Impagliazzo-Kabanets ’03) showed that a derandomized identity test would imply circuit lower bounds for NEXP.
- Thus, a derandomization of identity testing would both:
  - provide evidence that randomization in algorithms is dispensable, and
  - give circuit lower bounds.
Deeper Connections

• (Impagliazzo-Kabanets ’03) showed that a derandomized identity test would imply circuit lower bounds for NEXP.
• Thus, a derandomization of identity testing would both:
  • provide evidence that randomization in algorithms is dispensable, and
  • give circuit lower bounds.
Deeper Connections

• (Impagliazzo-Kabanets ’03) showed that a derandomized identity test would imply circuit lower bounds for NEXP.
• Thus, a derandomization of identity testing would both:
  • provide evidence that randomization in algorithms is dispensable, and
  • give circuit lower bounds.
Deeper Connections

- (Impagliazzo-Kabanets '03) showed that a derandomized identity test would imply circuit lower bounds for NEXP.
- Thus, a derandomization of identity testing would both:
  - provide evidence that randomization in algorithms is dispensable, and
  - give circuit lower bounds.
Outline

Motivation

Identity Testing

Constant Depth Circuits

Conclusion
Progress

- Some progress has been made when the input circuit has bounded many levels.
- Multilinear circuits of depth 3: (Raz-Shpilka ’04) gave a deterministic polynomial time identity test.
- Circuits of depth 3 with bounded top fanin: (Kayal-S ’06) gave a deterministic polynomial time identity test.
Progress

• Some progress has been made when the input circuit has bounded many levels.

• Multilinear circuits of depth 3: (Raz-Shpilka '04) gave a deterministic polynomial time identity test.

• Circuits of depth 3 with bounded top fanin: (Kayal-S '06) gave a deterministic polynomial time identity test.
Progress

- Some progress has been made when the input circuit has bounded many levels.
- Multilinear circuits of depth 3: (Raz-Shpilka '04) gave a deterministic polynomial time identity test.
- Circuits of depth 3 with bounded top fanin: (Kayal-S '06) gave a deterministic polynomial time identity test.
Depth 3 Circuits: The Setting

- For identity testing, it is sufficient to consider a "sum of product of linear functions" (ΣΠΣ circuit).
Depth 3 Circuits: The Setting

- For identity testing, it is sufficient to consider a “sum of product of linear functions” ($\Sigma \Pi \Sigma$ circuit).
Depth 3 Circuits: The Setting

Our input circuit $C$ over a field $\mathbb{F}$ will look like:

$$C(z_1, \ldots, z_n) = T_1 + \cdots + T_k$$

where $T_i$ is a product of linear functions $L_{i,1}, \ldots, L_{i,d}$

where $L_{i,j} = (a_{i,j,0} + a_{i,j,1}z_1 + \cdots + a_{i,j,n}z_n), a's \in \mathbb{F}$.
**Depth 3 Circuits: The Setting**

- Our input circuit $C$ over a field $\mathbb{F}$ will look like:
  
  $$C(z_1, \ldots, z_n) = T_1 + \cdots + T_k$$

  where $T_i$ is a product of linear functions $L_{i,1}, \ldots, L_{i,d}$
  
  where $L_{i,j} = (a_{i,j,0} + a_{i,j,1}z_1 + \cdots + a_{i,j,n}z_n)$, $a$'s $\in \mathbb{F}$. 
Depth 3 Circuits: The Setting

- Our input circuit $C$ over a field $\mathbb{F}$ will look like:
  $$C(z_1, \ldots, z_n) = T_1 + \cdots + T_k$$
  where $T_i$ is a product of linear functions $L_{i,1}, \ldots, L_{i,d}$
  where $L_{i,j} = (a_{i,j,0} + a_{i,j,1}z_1 + \cdots + a_{i,j,n}z_n)$, $a$'s $\in \mathbb{F}$. 
Our input circuit $C$ over a field $\mathbb{F}$ will look like:

$$C(z_1, \ldots, z_n) = T_1 + \cdots + T_k$$

where $T_i$ is a product of linear functions $L_{i,1}, \ldots, L_{i,d}$
where $L_{i,j} = (a_{i,j,0} + a_{i,j,1}z_1 + \cdots + a_{i,j,n}z_n)$, $a's \in \mathbb{F}$. 
The Idea of Chinese Remaindering

- Let $C$ be:

\[ C(x_1, \ldots, x_n) = T_1 + \cdots + T_k \]

where $T_i = L_{i,1} \cdots L_{i,d}$

- Pick $(d + 1)$ coprime linear functions $p_1, \ldots, p_{d+1}$ from the set \{\(L_{i,j} \mid i \in [k], j \in [d]\}\).

- $C = 0$ if and only if for all $i \in [d + 1]$, $C = 0 \pmod{p_i}$.

- $C \neq 0 \pmod{p_i}$ can be checked recursively because:
  - $C$ modulo $p_i$ has top fanin at most $(k - 1)$ because for some $j$, $T_j = 0 \pmod{p_i}$.
  - Let $\tau$ be an invertible map on $x_1, \ldots, x_n$ sending $p_i \mapsto x_1$.
  - Then $C = 0 \pmod{p_i}$ if and only if $C(\tau(x_1), \ldots, \tau(x_n)) = 0 \pmod{x_1}$.
The Idea of Chinese Remaindering

- Let $C$ be:
  \[ C(x_1, \ldots, x_n) = T_1 + \cdots + T_k \]
  where $T_i = L_{i,1} \cdots L_{i,d}$

- Pick $(d+1)$ coprime linear functions $p_1, \ldots, p_{d+1}$ from the set \{ $L_{i,j} \mid i \in [k], j \in [d]$ \}.

- $C = 0$ iff for all $i \in [d+1]$, $C = 0 \pmod{p_i}$.

- $C \neq 0 \pmod{p_i}$ can be checked recursively because:
  - $C$ modulo $p_i$ has top fanin at most $(k-1)$ because for some $j$, $T_j = 0 \pmod{p_i}$.
  - Let $\tau$ be an invertible map on $x_1, \ldots, x_n$ sending $p_i \mapsto x_1$.
  - Then $C = 0 \pmod{p_i}$ iff $C(\tau(x_1), \ldots, \tau(x_n)) = 0 \pmod{x_1}$.
The Idea of Chinese Remaindering

• Let $C$ be:

$$C(x_1, \ldots, x_n) = T_1 + \cdots + T_k$$

where $T_i = L_{i,1} \cdots L_{i,d}$

• Pick $(d + 1)$ coprime linear functions $p_1, \ldots, p_{d+1}$ from the set $\{L_{i,j} \mid i \in [k], j \in [d]\}$.

• $C = 0$ iff for all $i \in [d+1]$, $C = 0 \pmod{p_i}$.

• $C \not= 0 \pmod{p_i}$ can be checked recursively because:
  • $C$ modulo $p_i$ has top fanin at most $(k - 1)$ because for some $j$, $T_j = 0 \pmod{p_i}$.
  • Let $\tau$ be an invertible map on $x_1, \ldots, x_n$ sending $p_i \mapsto x_1$.
  • Then $C = 0 \pmod{p_i}$ iff $C(\tau(x_1), \ldots, \tau(x_n)) = 0 \pmod{x_1}$. 
The Idea of Chinese Remaindering

- Let $C$ be:
  
  \[ C(x_1, \ldots, x_n) = T_1 + \cdots + T_k \]
  
  where $T_i = L_{i,1} \cdots L_{i,d}$

- Pick $(d + 1)$ coprime linear functions $p_1, \ldots, p_{d+1}$ from the set $\{L_{i,j} \mid i \in [k], j \in [d]\}$.

- $C = 0$ iff for all $i \in [d + 1]$, $C = 0 \pmod{p_i}$.

- $C \neq 0 \pmod{p_i}$ can be checked recursively because:
  - $C$ modulo $p_i$ has top fanin at most $(k - 1)$ because for some $j$, $T_j = 0 \pmod{p_i}$.
  - Let $\tau$ be an invertible map on $x_1, \ldots, x_n$ sending $p_i \mapsto x_1$.
  - Then $C = 0 \pmod{p_i}$ iff $C(\tau(x_1), \ldots, \tau(x_n)) = 0 \pmod{x_1}$. 
The Idea of Chinese Remaindering

- Let $C$ be:
  \[ C(x_1, \ldots, x_n) = T_1 + \cdots + T_k \]
  where $T_i = L_{i,1} \cdots L_{i,d}$
- Pick $(d + 1)$ coprime linear functions $p_1, \ldots, p_{d+1}$ from the set $\{L_{i,j} \mid i \in [k], j \in [d]\}$.
- $C = 0$ iff for all $i \in [d+1]$, $C = 0 \pmod{p_i}$.
- $C \not\equiv 0 \pmod{p_i}$ can be checked recursively because:
  - $C$ modulo $p_i$ has top fanin atmost $(k - 1)$ because for some $j$, $T_j \equiv 0 \pmod{p_i}$.
  - Let $\tau$ be an invertible map on $x_1, \ldots, x_n$ sending $p_i \mapsto x_1$.
  - Then $C = 0 \pmod{p_i}$ iff $C(\tau(x_1), \ldots, \tau(x_n)) = 0 \pmod{x_1}$. 
The Idea of Chinese Remaindering

- Let $C$ be:
  $$C(x_1, \ldots, x_n) = T_1 + \cdots + T_k$$
  where $T_i = L_{i,1} \cdots L_{i,d}$

- Pick $(d + 1)$ coprime linear functions $p_1, \ldots, p_{d+1}$ from the set \{\(L_{i,j} \mid i \in [k], j \in [d]\)\}.

- $C = 0$ iff for all $i \in [d + 1]$, $C = 0 \pmod{p_i}$.

- $C \neq 0 \pmod{p_i}$ can be checked recursively because:
  - $C$ modulo $p_i$ has top fanin atmost $(k - 1)$ because for some $j$, $T_j = 0 \pmod{p_i}$.
  - Let $\tau$ be an invertible map on $x_1, \ldots, x_n$ sending $p_i \mapsto x_1$.
  - Then $C = 0 \pmod{p_i}$ iff $C(\tau(x_1), \ldots, \tau(x_n)) = 0 \pmod{x_1}$. 
The Idea of Chinese Remaindering

- Let $C$ be:

$$C(x_1, \ldots, x_n) = T_1 + \cdots + T_k$$

where $T_i = L_{i,1} \cdots L_{i,d}$

- Pick $(d + 1)$ coprime linear functions $p_1, \ldots, p_{d+1}$ from the set $\{L_{i,j} \mid i \in [k], j \in [d]\}$.

- $C = 0$ iff for all $i \in [d + 1]$, $C \equiv 0 \pmod{p_i}$.

- $C \not\equiv 0 \pmod{p_i}$ can be checked recursively because:
  - $C$ modulo $p_i$ has top fanin at most $(k - 1)$ because for some $j$, $T_j \equiv 0 \pmod{p_i}$.
  - Let $\tau$ be an invertible map on $x_1, \ldots, x_n$ sending $p_i \mapsto x_1$.
  - Then $C \equiv 0 \pmod{p_i}$ iff $C(\tau(x_1), \ldots, \tau(x_n)) \equiv 0 \pmod{x_1}$. 
The Idea of Chinese Remaindering

• Let \( C \) be:

\[
C(x_1, \ldots, x_n) = T_1 + \cdots + T_k
\]
where \( T_i = L_{i,1} \cdots L_{i,d} \)

• Pick \((d + 1)\) coprime linear functions \( p_1, \ldots, p_{d+1} \) from the set \( \{L_{i,j} \mid i \in [k], j \in [d]\} \).

• \( C = 0 \) iff for all \( i \in [d + 1], \ C = 0 \pmod{p_i}. \)

• \( C \neq 0 \pmod{p_i} \) can be checked recursively because:
  • \( C \) modulo \( p_i \) has top fanin atmost \((k - 1)\)
    because for some \( j, \ T_j = 0 \pmod{p_i}. \)
  • Let \( \tau \) be an invertible map on \( x_1, \ldots, x_n \) sending \( p_i \mapsto x_1. \)
  • Then \( C = 0 \pmod{p_i} \) iff \( C(\tau(x_1), \ldots, \tau(x_n)) = 0 \pmod{x_1}. \)
The Idea of Chinese Remaindering

• Let $C$ be:

$$C(x_1, \ldots, x_n) = T_1 + \cdots + T_k$$

where $T_i = L_{i,1} \cdots L_{i,d}$

• Pick $(d + 1)$ coprime linear functions $p_1, \ldots, p_{d+1}$ from the set $\{L_{i,j} \mid i \in [k], j \in [d]\}$.

• $C = 0$ iff for all $i \in [d + 1]$, $C = 0 \pmod{p_i}$.

• $C \not\equiv 0 \pmod{p_i}$ can be checked recursively because:
  
  • $C$ modulo $p_i$ has top fanin at most $(k - 1)$ because for some $j$, $T_j = 0 \pmod{p_i}$.
  
  • Let $\tau$ be an invertible map on $x_1, \ldots, x_n$ sending $p_i \mapsto x_1$.

  • Then $C = 0 \pmod{p_i}$ iff $C(\tau(x_1), \ldots, \tau(x_n)) = 0 \pmod{x_1}$. 
THE IDEA OF CHINESE REMAINDERING

- Let $C$ be:

$$C(x_1, \ldots, x_n) = T_1 + \cdots + T_k$$
where $T_i = L_{i,1} \cdots L_{i,d}$

- Pick $(d+1)$ coprime linear functions $p_1, \ldots, p_{d+1}$ from the set $\{L_{i,j} \mid i \in [k], j \in [d]\}$.

- $C = 0$ iff for all $i \in [d+1]$, $C = 0 \pmod{p_i}$.

- $C \equiv 0 \pmod{p_i}$ can be checked recursively because:
  - $C$ modulo $p_i$ has top fanin atmost $(k-1)$ because for some $j$, $T_j = 0 \pmod{p_i}$.
  - Let $\tau$ be an invertible map on $x_1, \ldots, x_n$ sending $p_i \mapsto x_1$.
  - Then $C = 0 \pmod{p_i}$ iff $C(\tau(x_1), \ldots, \tau(x_n)) = 0 \pmod{x_1}$. 
Chinese Remaindering needs generalization

- There may not always be $(d + 1)$ coprime linear functions in the set $\{L_{i,j} \mid i \in [k], j \in [d]\}$.
- So we need to pick powers $p_1^{e_1}, \ldots, p_\ell^{e_\ell}$ of coprime linear functions $p_1, \ldots, p_\ell$ such that,
  1. every $p_i^{e_i}$ divides some $T_j$.
  2. $e_1 + \cdots + e_\ell \geq d$.
- How do we check $C \equiv 0 \pmod{p_i^{e_i}}$?
- We transform $p_j \mapsto x_1$ by applying an invertible map $\tau$ on $x_1, \ldots, x_n$. Then $C = 0 \pmod{p_i^{e_i}}$ iff
  \[ C(\tau(x_1), \ldots, \tau(x_n)) = 0 \text{ over } \mathbb{F}[x_1]/(x_1^{e_i}). \]
- Thus, we recursively solve identity testing over “bigger” rings.
Chinese Remaindering needs generalization

- There may not always be \((d + 1)\) coprime linear functions in the set \(\{L_{i,j} \mid i \in [k], j \in [d]\}\).
- So we need to pick powers \(p_1^{e_1}, \ldots, p_\ell^{e_\ell}\) of coprime linear functions \(p_1, \ldots, p_\ell\) such that,
  1. every \(p_i^{e_i}\) divides some \(T_j\).
  2. \(e_1 + \cdots + e_\ell \geq d\).
- How do we check \(C \equiv 0 \pmod{p_i^{e_i}}\)?
- We transform \(p_j \mapsto x_1\) by applying an invertible map \(\tau\) on \(x_1, \ldots, x_n\). Then \(C = 0 \pmod{p_i^{e_i}}\) iff
  \[
  C(\tau(x_1), \ldots, \tau(x_n)) = 0 \text{ over } \mathbb{F}[x_1]/(x_1^{e_i}).
  \]
- Thus, we recursively solve identity testing over “bigger” rings.

Skip details
Chinese Remaindering needs generalization

- There may not always be \((d + 1)\) coprime linear functions in the set \(\{L_{i,j} \mid i \in [k], j \in [d]\}\).
- So we need to pick powers \(p_1^{e_1}, \ldots, p_\ell^{e_\ell}\) of coprime linear functions \(p_1, \ldots, p_\ell\) such that,
  1. every \(p_i^{e_i}\) divides some \(T_j\).
  2. \(e_1 + \cdots + e_\ell \geq d\).
- How do we check \(C \equiv 0 \pmod{p_i^{e_i}}\)?
- We transform \(p_i \mapsto x_1\) by applying an invertible map \(\tau\) on \(x_1, \ldots, x_n\). Then \(C \equiv 0 \pmod{p_i^{e_i}}\) iff
  \[C(\tau(x_1), \ldots, \tau(x_n)) = 0\] over \(\mathbb{F}[x_1]/(x_1^{e_i})\).
- Thus, we recursively solve identity testing over “bigger” rings.
Chinese Remaindering needs generalization

- There may not always be \((d + 1)\) coprime linear functions in the set \(\{L_{i,j} \mid i \in [k], j \in [d]\}\).
- So we need to pick powers \(p_1^{e_1}, \ldots, p_\ell^{e_\ell}\) of coprime linear functions \(p_1, \ldots, p_\ell\) such that,
  1. every \(p_i^{e_i}\) divides some \(T_j\).
  2. \(e_1 + \cdots + e_\ell \geq d\).

- How do we check \(C \equiv 0 \pmod{p_i^{e_i}}\)?
- We transform \(p_i \mapsto x_1\) by applying an invertible map \(\tau\) on \(x_1, \ldots, x_n\). Then \(C \equiv 0 \pmod{p_i^{e_i}}\) iff
  \[C(\tau(x_1), \ldots, \tau(x_n)) = 0\] over \(\mathbb{F}[x_1]/(x_1^{e_i})\).
- Thus, we recursively solve identity testing over “bigger” rings.
Chinese Remaindering needs generalization

- There may not always be \((d + 1)\) coprime linear functions in the set \(\{L_{i,j} \mid i \in [k], j \in [d]\}\).
- So we need to pick powers \(p_1^{e_1}, \ldots, p_\ell^{e_\ell}\) of coprime linear functions \(p_1, \ldots, p_\ell\) such that,
  1. every \(p_i^{e_i}\) divides some \(T_j\).
  2. \(e_1 + \cdots + e_\ell \geq d\).
- How do we check \(C \equiv 0 \pmod{p_i^{e_i}}\)?
- We transform \(p_i \mapsto x_1\) by applying an invertible map \(\tau\) on \(x_1, \ldots, x_n\). Then \(C \equiv 0 \pmod{p_i^{e_i}}\) iff
  \[C(\tau(x_1), \ldots, \tau(x_n)) = 0\] over \(\mathbb{F}[x_1]/(x_1^{e_i})\).
- Thus, we recursively solve identity testing over “bigger” rings.
Chinese Remaindering needs generalization

- There may not always be \((d + 1)\) coprime linear functions in the set \(\{L_{i,j} \mid i \in [k], j \in [d]\}\).
- So we need to pick powers \(p_1^{e_1}, \ldots, p_{\ell}^{e_{\ell}}\) of coprime linear functions \(p_1, \ldots, p_{\ell}\) such that,
  1. every \(p_i^{e_i}\) divides some \(T_j\).
  2. \(e_1 + \cdots + e_{\ell} \geq d\).
- How do we check \(C \equiv 0 \pmod{p_i^{e_i}}\)?
- We transform \(p_i \mapsto x_1\) by applying an invertible map \(\tau\) on \(x_1, \ldots, x_n\). Then \(C \equiv 0 \pmod{p_i^{e_i}}\) iff
  \[C(\tau(x_1), \ldots, \tau(x_n)) = 0\] over \(\mathbb{F}[x_1]/(x_1^{e_i})\).
- Thus, we recursively solve identity testing over “bigger” rings.
There may not always be $(d + 1)$ coprime linear functions in the set $\{L_{i,j} | i \in [k], j \in [d]\}$.

So we need to pick powers $p_1^{e_1}, \ldots, p_\ell^{e_\ell}$ of coprime linear functions $p_1, \ldots, p_\ell$ such that,

1. every $p_i^{e_i}$ divides some $T_j$.
2. $e_1 + \cdots + e_\ell \geq d$.

How do we check $C \equiv 0 \pmod{p_i^{e_i}}$?

We transform $p_i \mapsto x_1$ by applying an invertible map $\tau$ on $x_1, \ldots, x_n$. Then $C \equiv 0 \pmod{p_i^{e_i}}$ iff

$$C(\tau(x_1), \ldots, \tau(x_n)) = 0 \text{ over } \mathbb{F}[x_1]/(x_1^{e_i}).$$

Thus, we recursively solve identity testing over “bigger” rings.
**Chinese Remaindering needs generalization**

- There may not always be \((d + 1)\) coprime linear functions in the set \(\{L_{i,j} \mid i \in [k], j \in [d]\}\).

- So we need to pick powers \(p_{1}^{e_{1}}, \ldots, p_{\ell}^{e_{\ell}}\) of coprime linear functions \(p_{1}, \ldots, p_{\ell}\) such that,
  1. every \(p_{i}^{e_{i}}\) divides some \(T_{j}\).
  2. \(e_{1} + \cdots + e_{\ell} \geq d\).

- How do we check \(C \equiv 0 \pmod{p_{i}^{e_{i}}})?\)

- We transform \(p_{i} \mapsto x_{1}\) by applying an invertible map \(\tau\) on \(x_{1}, \ldots, x_{n}\). Then \(C \equiv 0 \pmod{p_{i}^{e_{i}}}\) iff
  \[
  C(\tau(x_{1}), \ldots, \tau(x_{n})) = 0 \text{ over } \mathbb{F}[x_{1}]/(x_{1}^{e_{i}}).
  \]

- Thus, we recursively solve identity testing over “bigger” rings.
Chinese Remaindering needs generalization

- There may not always be \((d + 1)\) coprime linear functions in the set \(\{L_{i,j} \mid i \in [k], j \in [d]\}\).
- So we need to pick powers \(p_1^{e_1}, \ldots, p_\ell^{e_\ell}\) of coprime linear functions \(p_1, \ldots, p_\ell\) such that,
  1. every \(p_i^{e_i}\) divides some \(T_j\).
  2. \(e_1 + \cdots + e_\ell \geq d\).

- How do we check \(C \equiv 0 \pmod{p_i^{e_i}}\)?
- We transform \(p_i \mapsto x_1\) by applying an invertible map \(\tau\) on \(x_1, \ldots, x_n\). Then \(C \equiv 0 \pmod{p_i^{e_i}}\) \(\iff\)
  \[C(\tau(x_1), \ldots, \tau(x_n)) = 0\] over \(\mathbb{F}[x_1]/(x_1^{e_i})\).
- Thus, we recursively solve identity testing over “bigger” rings.

Skip details
Identity Test (in more detail)

- Let $R$ be a local subring of $\mathbb{F}[x_1, \ldots, x_m]$ with maximal ideal $\mathcal{M}$.
- Let the input be a $\Sigma \Pi \Sigma$ circuit $C(z_1, \ldots, z_n)$ in $R[z_1, \ldots, z_n]$:
  
  $C = T_1 + \cdots + T_k$

  where, $T_i = L_{i,1} \cdots L_{i,d}$

- Wlog let $T_1$ produce the lexicographically largest monomial.

- $T_1$ can be factored into coprime polynomials as follows:
  
  $T_1 = \alpha \cdot p_1(z_1, \ldots, z_n) \cdots p_s(z_1, \ldots, z_n)$

  where, $p_i = (\ell_{i} + m_{i,1}) \cdots (\ell_{i} + m_{i,d_i})$ for some linear form $\ell_i$ and $\alpha$, $m_{i,j}$'s are in $\mathcal{M}$. 

Identity Test (in more detail)

- Let $R$ be a local subring of $\mathbb{F}[x_1, \ldots, x_m]$ with maximal ideal $M$.
- Let the input be a $\Sigma \Pi \Sigma$ circuit $C(z_1, \ldots, z_n)$ in $R[z_1, \ldots, z_n]$: $C = T_1 + \cdots + T_k$ where, $T_i = L_{i,1} \cdots L_{i,d}$
- Wlog let $T_1$ produce the lexicographically largest monomial.
- $T_1$ can be factored into coprime polynomials as follows: $T_1 = \alpha \cdot p_1(z_1, \ldots, z_n) \cdots p_s(z_1, \ldots, z_n)$ where, $p_i = (\ell_i + m_{i,1}) \cdots (\ell_i + m_{i,d_i})$ for some linear form $\ell_i$ and $\alpha$, $m_{i,j}$’s are in $M$. 
Identity Test (in more detail)

- Let $R$ be a local subring of $\mathbb{F}[x_1, \ldots, x_m]$ with maximal ideal $\mathcal{M}$.
- Let the input be a $\Sigma\Pi\Sigma$ circuit $C(z_1, \ldots, z_n)$ in $R[z_1, \ldots, z_n]$:
  $C = T_1 + \cdots + T_k$
  where, $T_i = L_{i,1} \cdots L_{i,d}$
- Wlog let $T_1$ produce the lexicographically largest monomial.
- $T_1$ can be factored into coprime polynomials as follows:
  $T_1 = \alpha \cdot p_1(z_1, \ldots, z_n) \cdots p_s(z_1, \ldots, z_n)$
  where, $p_i = (\ell_i + m_{i,1}) \cdots (\ell_i + m_{i,d_i})$ for some linear form $\ell_i$ and $\alpha$, $m_{i,j}$'s are in $\mathcal{M}$. 
Identity Test (in more detail)

- Let $R$ be a local subring of $\mathbb{F}[x_1, \ldots, x_m]$ with maximal ideal $\mathcal{M}$.
- Let the input be a $\Sigma\Pi\Sigma$ circuit $C(z_1, \ldots, z_n)$ in $R[z_1, \ldots, z_n]$:
  $C = T_1 + \cdots + T_k$
  where, $T_i = L_{i,1} \cdots L_{i,d}$
- Wlog let $T_1$ produce the lexicographically largest monomial.
- $T_1$ can be factored into coprime polynomials as follows:
  $T_1 = \alpha \cdot p_1(z_1, \ldots, z_n) \cdots p_s(z_1, \ldots, z_n)$
  where, $p_i = (\ell_i + m_{i,1}) \cdots (\ell_i + m_{i,d_i})$ for some linear form $\ell_i$ and $\alpha$, $m_{i,j}$'s are in $\mathcal{M}$. 
Identity Test (in more detail)

- Let $R$ be a local subring of $\mathbb{F}[x_1, \ldots, x_m]$ with maximal ideal $\mathcal{M}$.
- Let the input be a $\Sigma \Pi \Sigma$ circuit $C(z_1, \ldots, z_n)$ in $R[z_1, \ldots, z_n]$:
  \[ C = T_1 + \cdots + T_k \]
  where, $T_i = L_{i,1} \cdots L_{i,d}$
- Wlog let $T_1$ produce the lexicographically largest monomial.
- $T_1$ can be factored into coprime polynomials as follows:
  \[ T_1 = \alpha \cdot p_1(z_1, \ldots, z_n) \cdots p_s(z_1, \ldots, z_n) \]
  where, $p_i = (\ell_i + m_{i,1}) \cdots (\ell_i + m_{i,d_i})$ for some linear form $\ell_i$ and $\alpha$, $m_{i,j}$'s are in $\mathcal{M}$. 
**Identity Test (in more detail)**

- Let $R$ be a local subring of $\mathbb{F}[x_1, \ldots, x_m]$ with maximal ideal $\mathcal{M}$.
- Let the input be a $\Sigma \Pi \Sigma$ circuit $C(z_1, \ldots, z_n)$ in $R[z_1, \ldots, z_n]$: $C = T_1 + \cdots + T_k$ where, $T_i = L_{i,1} \cdots L_{i,d}$
- Wlog let $T_1$ produce the lexicographically largest monomial.
- $T_1$ can be factored into *coprime* polynomials as follows: $T_1 = \alpha \cdot p_1(z_1, \ldots, z_n) \cdots p_s(z_1, \ldots, z_n)$ where, $p_i = (\ell_i + m_{i,1}) \cdots (\ell_i + m_{i,d_i})$ for some linear form $\ell_i$ and $\alpha$, $m_{i,j}$'s are in $\mathcal{M}$. 
Identity Test (in more detail)

- Let $R$ be a local subring of $\mathbb{F}[x_1, \ldots, x_m]$ with maximal ideal $\mathcal{M}$.
- Let the input be a $\Sigma \Pi \Sigma$ circuit $C(z_1, \ldots, z_n)$ in $R[z_1, \ldots, z_n]$:
  $C = T_1 + \cdots + T_k$
  where, $T_i = L_{i,1} \cdots L_{i,d}$
- Wlog let $T_1$ produce the lexicographically largest monomial.
- $T_1$ can be factored into coprime polynomials as follows:
  $T_1 = \alpha \cdot p_1(z_1, \ldots, z_n) \cdots p_s(z_1, \ldots, z_n)$
  where, $p_i = (\ell_i + m_{i,1}) \cdots (\ell_i + m_{i,d_i})$ for some linear form $\ell_i$
  and $\alpha, m_{i,j}$'s are in $\mathcal{M}$. 
Identity Test (in more detail)

• Let $R$ be a local subring of $\mathbb{F}[x_1, \ldots, x_m]$ with maximal ideal $M$.

• Let the input be a $\Sigma\Pi\Sigma$ circuit $C(z_1, \ldots, z_n)$ in $R[z_1, \ldots, z_n]$:

$$C = T_1 + \cdots + T_k$$

where, $T_i = L_{i,1} \cdots L_{i,d}$

• Wlog let $T_1$ produce the lexicographically largest monomial.

• $T_1$ can be factored into coprime polynomials as follows:

$$T_1 = \alpha \cdot p_1(z_1, \ldots, z_n) \cdots p_s(z_1, \ldots, z_n)$$

where, $p_i = (\ell_i + m_{i,1}) \cdots (\ell_i + m_{i,d_i})$ for some linear form $\ell_i$ and $\alpha, m_{i,j}$’s are in $M$. 
Identity Test (in more detail)

- $C(z_1, \ldots, z_n) = 0$ iff for all $i \in [s]$, $C = 0 \pmod{p_i}$ and lexicographically largest monomial of $C$ has zero coefficient.

- For a fixed $i$: transform $l_i \mapsto z_1$ by an invertible linear transformation $\tau_i$ on $z_1, \ldots, z_n$ and, thus, $p_i \mapsto (z_1 + m_{i,1}) \cdots (z_1 + m_{i,d_i})$.

- Then $C = 0 \pmod{p_i}$ iff $\tau_i(C) = 0 \pmod{(z_1 + m_{i,1}) \cdots (z_1 + m_{i,d_i})}$.

- This entails checking $\tau_i(T_2) + \cdots + \tau_i(T_k) = 0$ over the local ring $R[z_1]/((z_1 + m_{i,1}) \cdots (z_1 + m_{i,d_i}))$.

- Thus, we can recursively check whether $C = 0 \pmod{p_i}$. 
Identity Test (in more detail)

• $C(z_1, \ldots, z_n) = 0$ iff

  for all $i \in [s]$, $C = 0 \pmod{p_i}$

  and

  lexicographically largest monomial of $C$ has zero coefficient.

• For a fixed $i$: transform $\ell_i \mapsto z_1$ by an invertible linear transformation $\tau_i$ on $z_1, \ldots, z_n$ and, thus,

  $p_i \mapsto (z_1 + m_i,1) \cdots (z_1 + m_i,d_i)$

• Then $C = 0 \pmod{p_i}$ iff

  $\tau_i(C) = 0 \pmod{(z_1 + m_i,1) \cdots (z_1 + m_i,d_i)}$.

• This entails checking $\tau_i(T_2) + \cdots + \tau_i(T_k) = 0$ over the local ring $R[z_1]/((z_1 + m_i,1) \cdots (z_1 + m_i,d_i))$.

• Thus, we can recursively check whether $C = 0 \pmod{p_i}$. 
Constant Depth Circuits

Identity Test (in more detail)

- \( C(z_1, \ldots, z_n) = 0 \) iff for all \( i \in [s] \), \( C = 0 \) (mod \( p_i \)) and lexicographically largest monomial of \( C \) has zero coefficient.

- For a fixed \( i \): transform \( \ell_i \mapsto z_1 \) by an invertible linear transformation \( \tau_i \) on \( z_1, \ldots, z_n \) and, thus, \( p_i \mapsto (z_1 + m_i, 1) \cdots (z_1 + m_i, d_i) \).

- Then \( C = 0 \) (mod \( p_i \)) iff \( \tau_i(C) = 0 \) (mod \( (z_1 + m_i, 1) \cdots (z_1 + m_i, d_i) \)).

- This entails checking \( \tau_i(T_2) + \cdots + \tau_i(T_k) = 0 \) over the local ring \( R[z_1]/((z_1 + m_i, 1) \cdots (z_1 + m_i, d_i)) \).

- Thus, we can recursively check whether \( C = 0 \) (mod \( p_i \)).
Identity Test (in more detail)

- $C(z_1, \ldots, z_n) = 0$ iff for all $i \in \{s\}$, $C = 0 \pmod{p_i}$ and lexicographically largest monomial of $C$ has zero coefficient.

- For a fixed $i$: transform $\ell_i \mapsto z_1$ by an invertible linear transformation $\tau_i$ on $z_1, \ldots, z_n$ and, thus, $p_i \mapsto (z_1 + m_{i,1}) \cdots (z_1 + m_{i,d_i})$.

- Then $C = 0 \pmod{p_i}$ iff $\tau_i(C) = 0 \pmod{(z_1 + m_{i,1}) \cdots (z_1 + m_{i,d_i})}$.

- This entails checking $\tau_i(T_2) + \cdots + \tau_i(T_k) = 0$ over the local ring $R[z_1]/((z_1 + m_{i,1}) \cdots (z_1 + m_{i,d_i}))$.

- Thus, we can recursively check whether $C = 0 \pmod{p_i}$. 


Identity Test (in more detail)

- \( C(z_1, \ldots, z_n) = 0 \) iff
  
  for all \( i \in [s] \), \( C = 0 \) (mod \( p_i \))
  
  and

  lexicographically largest monomial of \( C \) has zero coefficient.

- For a fixed \( i \): transform \( \ell_i \mapsto z_1 \) by an invertible linear transformation \( \tau_i \) on \( z_1, \ldots, z_n \) and, thus,

  \( p_i \mapsto (z_1 + m_{i,1}) \cdots (z_1 + m_{i,d_i}) \)

- Then \( C = 0 \) (mod \( p_i \)) iff

  \( \tau_i(C) = 0 \) (mod \((z_1 + m_{i,1}) \cdots (z_1 + m_{i,d_i}))\).

- This entails checking \( \tau_i(T_2) + \cdots + \tau_i(T_k) = 0 \) over the local ring \( R[z_1]/((z_1 + m_{i,1}) \cdots (z_1 + m_{i,d_i})) \).

- Thus, we can recursively check whether \( C = 0 \) (mod \( p_i \)).


**Identity Test (in more detail)**

- \( C(z_1, \ldots, z_n) = 0 \) iff
  
  for all \( i \in [s] \), \( C = 0 \) (mod \( p_i \))
  
  and
  
  lexicographically largest monomial of \( C \) has zero coefficient.

- For a fixed \( i \): transform \( \ell_i \mapsto z_1 \) by an invertible linear transformation \( \tau_i \) on \( z_1, \ldots, z_n \) and, thus,
  
  \( p_i \mapsto (z_1 + m_{i,1}) \cdots (z_1 + m_{i,d_i}) \)

- Then \( C = 0 \) (mod \( p_i \)) iff
  
  \( \tau_i(C) = 0 \) (mod \( (z_1 + m_{i,1}) \cdots (z_1 + m_{i,d_i})) \).

- This entails checking \( \tau_i(T_2) + \cdots + \tau_i(T_k) = 0 \) over the local ring \( R[z_1]/((z_1 + m_{i,1}) \cdots (z_1 + m_{i,d_i})) \).

- Thus, we can recursively check whether \( C = 0 \) (mod \( p_i \)).
Time Complexity

- Note that in each recursive call:
  1. Fanin $k$ reduces by at least 1
  2. Dimension of the base ring increases at most $d$ times.

- The computations that we do are on rings of dimension at most $d^k$.

- Identity testing for depth 3 circuits over $n$ variables, total degree $d$ and top fanin $k$ can be done in time $\text{poly}(d^k, n)$. 
**Time Complexity**

- Note that in each recursive call:
  1. Fanin $k$ reduces by at least 1
  2. Dimension of the base ring increases at most $d$ times.

- The computations that we do are on rings of dimension at most $d^k$.

- Identity testing for depth 3 circuits over $n$ variables, total degree $d$ and top fanin $k$ can be done in time $poly(d^k, n)$. 
Time Complexity

- Note that in each recursive call:
  1. Fanin $k$ reduces by at least 1
  2. Dimension of the base ring increases at most $d$ times.
- The computations that we do are on rings of dimension at most $d^k$.
- Identity testing for depth 3 circuits over $n$ variables, total degree $d$ and top fanin $k$ can be done in time $\text{poly}(d^k, n)$. 
**Time Complexity**

- Note that in each recursive call:
  1. Fanin $k$ reduces by at least 1
  2. Dimension of the base ring increases at most $d$ times.
- The computations that we do are on rings of dimension at most $d^k$.
- Identity testing for depth 3 circuits over $n$ variables, total degree $d$ and top fanin $k$ can be done in time $\text{poly}(d^k, n)$. 
Time Complexity

- Note that in each recursive call:
  1. Fanin $k$ reduces by at least 1
  2. Dimension of the base ring increases at most $d$ times.
- The computations that we do are on rings of dimension at most $d^k$.
- Identity testing for depth 3 circuits over $n$ variables, total degree $d$ and top fanin $k$ can be done in time $\text{poly}(d^k, n)$.
Conclusion

Outline

Motivation

Identity Testing

Constant Depth Circuits

Conclusion
In Conclusion

- Depth 3 Identity testing for bounded top fanin is in P.
- Open Problem: Identity testing for general depth 3 circuits?
- "Easier" Open Problem: Identity testing for a diagonalized $\Sigma \Pi \Sigma$ circuit, i.e.,

$$C(x_1, \ldots, x_n) = L_1^d + \cdots + L_k^d$$

where, $L_1, \ldots, L_k$ are linear functions.

Questions?
In Conclusion

- Depth 3 Identity testing for bounded top fanin is in P.
- Open Problem: Identity testing for general depth 3 circuits?
- “Easier” Open Problem: Identity testing for a diagonalized $\Sigma \Pi \Sigma$ circuit, i.e.,

$$C(x_1, \ldots, x_n) = L_1^d + \cdots + L_k^d$$

where, $L_1, \ldots, L_k$ are linear functions.

Questions?
In Conclusion

- Depth 3 Identity testing for bounded top fanin is in P.
- Open Problem: Identity testing for general depth 3 circuits?
- “Easier” Open Problem: Identity testing for a diagonalized \( \Sigma \Pi \Sigma \) circuit, i.e.,

\[
C(x_1, \ldots, x_n) = L_1^d + \cdots + L_k^d
\]

where, \( L_1, \ldots, L_k \) are linear functions.

Questions?
In Conclusion

- Depth 3 Identity testing for bounded top fanin is in P.
- Open Problem: Identity testing for general depth 3 circuits?
- "Easier" Open Problem: Identity testing for a \textit{diagonalized} \(\Sigma\Pi\Sigma\) circuit, i.e.,

\[
C(x_1, \ldots, x_n) = L_1^d + \cdots + L_k^d
\]

where, \(L_1, \ldots, L_k\) are linear functions.

Questions?
In Conclusion

- Depth 3 Identity testing for bounded top fanin is in P.
- Open Problem: Identity testing for general depth 3 circuits?
- "Easier" Open Problem: Identity testing for a diagonalized $\Sigma\Pi\Sigma$ circuit, i.e.,

$$C(x_1, \ldots, x_n) = L_1^d + \cdots + L_k^d$$

where, $L_1, \ldots, L_k$ are linear functions.

Questions?
In Conclusion

- Depth 3 Identity testing for bounded top fanin is in P.
- Open Problem: Identity testing for general depth 3 circuits?
- “Easier” Open Problem: Identity testing for a diagonalized $\Sigma\Pi\Sigma$ circuit, i.e.,

$$C(x_1, \ldots, x_n) = L_1^d + \cdots + L_k^d$$

where, $L_1, \ldots, L_k$ are linear functions.

Questions?