Complete Derandomization of Identity Testing of Read-Once Formulas

Daniel Minahan Ilya Volkovich

University of Michigan

Presented by: Ramprasad Saptharishi

Outline

Outline

Outline

Outline

Outline

Problem Definition

Definition (Polynomial Identity Testing (PIT))

Given a succinct representation of a polynomial P in n variables over a field $\mathbb{F}$, decide efficiently if P is identically 0.

Problem Definition

Definition (Polynomial Identity Testing (PIT))

Given a succinct representation of a polynomial P in n variables over a field $\mathbb{F}$, decide efficiently if P is identically 0.

This requires a few things to be specified.

Problem Definition

Definition (Polynomial Identity Testing (PIT))

Given a succinct representation of a polynomial P in n variables over a field $\mathbb{F}$, decide efficiently if P is identically 0.

This requires a few things to be specified.

- Method of representation

Problem Definition

Definition (Polynomial Identity Testing (PIT))

Given a succinct representation of a polynomial P in n variables over a field $\mathbb{F}$, decide efficiently if P is identically 0.

This requires a few things to be specified.

- Method of representation
- Efficiency

Problem Definition

Definition (Polynomial Identity Testing (PIT))

Given a succinct representation of a polynomial P in n variables over a field $\mathbb{F}$, decide efficiently if P is identically 0.

This requires a few things to be specified.

- Method of representation
- Efficiency
- Allowed actions

Succinct Representation

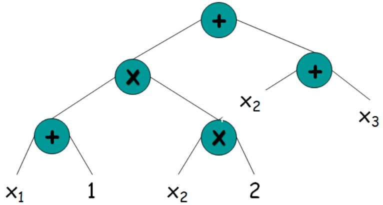
Succinct Representation

Definition

An arithmetic formula is a directed tree from variables $(x_1, \dots, x_n)$ to an output. Each leaf is assigned a variable or field element and each internal node, or gate, is assigned an operation $+$ or $\times$.

Idea: Model small computational devices

Figure: Example of an Arithmetic Formula for $(x_1 + 1)(2x_2) + x_2 + x_3$



Black-Box Setting

- We cannot see the formula.

Black-Box Setting

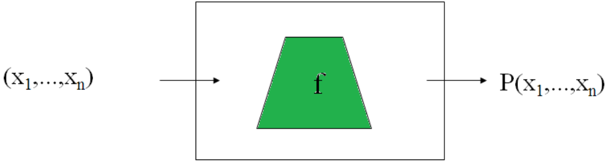
- We cannot see the formula.
- We can only send inputs to the formula and receive an output.

Black-Box Setting

- We cannot see the formula.
- We can only send inputs to the formula and receive an output.

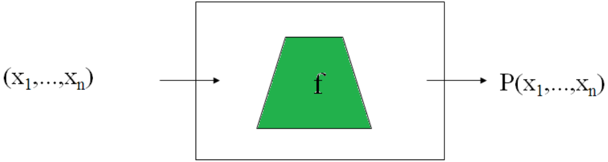
Black-Box Setting

- We cannot see the formula.
- We can only send inputs to the formula and receive an output.



Black-Box Setting

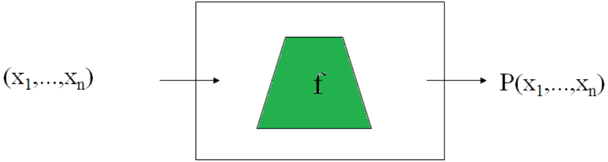
- We cannot see the formula.
- We can only send inputs to the formula and receive an output.



- We are allowed to query the formula on a small extension field.

Black-Box Setting

- We cannot see the formula.
- We can only send inputs to the formula and receive an output.



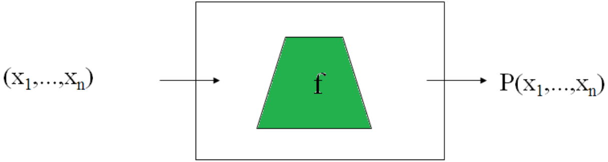
- We are allowed to query the formula on a small extension field.

Definition (Hitting Set)

Let $\mathcal{C} \subseteq \mathbb{F}[x_1, \ldots, x_n]$. A set $\mathcal{H} \subseteq \mathbb{F}^n$ is called a *hitting set* for $\mathcal{C}$ provided that for any $P \in \mathcal{C}$ with $P \not\equiv 0$, $\exists \bar{a} \in \mathcal{H}$ with $P(\bar{a}) \neq 0$.

Black-Box Setting

- We cannot see the formula.
- We can only send inputs to the formula and receive an output.



- We are allowed to query the formula on a small extension field.

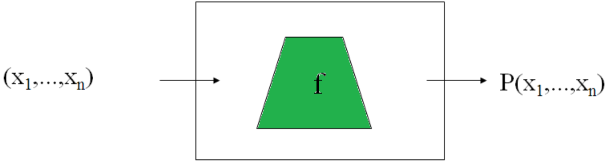
Definition (Hitting Set)

Let $\mathcal{C} \subseteq \mathbb{F}[x_1, \ldots, x_n]$. A set $\mathcal{H} \subseteq \mathbb{F}^n$ is called a *hitting set* for $\mathcal{C}$ provided that for any $P \in \mathcal{C}$ with $P \not\equiv 0$, $\exists \bar{a} \in \mathcal{H}$ with $P(\bar{a}) \neq 0$.

- $\mathcal{H}$ gives an algorithm that runs in time $O(|\mathcal{H}|)$.

Black-Box Setting

- We cannot see the formula.
- We can only send inputs to the formula and receive an output.



- We are allowed to query the formula on a small extension field.

Definition (Hitting Set)

Let $\mathcal{C} \subseteq \mathbb{F}[x_1,\ldots,x_n]$. A set $\mathcal{H} \subseteq \mathbb{F}^n$ is called a *hitting set* for $\mathcal{C}$ provided that for any $P \in \mathcal{C}$ with $P \not\equiv 0$, $\exists \bar{a} \in \mathcal{H}$ with $P(\bar{a}) \neq 0$.

- $\mathcal{H}$ gives an algorithm that runs in time $O(|\mathcal{H}|)$.
- Goal: find a small hitting set for polynomials computed by small formulas.

Applications

Various application in Complexity Theory and Algorithm Design:

Applications

Various application in Complexity Theory and Algorithm Design:

- Fibonacci Identity:
  $(a^2 + b^2)(c^2 + d^2) = (ac - bd)^2 + (ad + bc)^2$

Applications

Various application in Complexity Theory and Algorithm Design:

- Fibonacci Identity:
  $(a^2 + b^2)(c^2 + d^2) = (ac - bd)^2 + (ad + bc)^2$
- Graph Perfect Matching [MVV87]

Applications

Various application in Complexity Theory and Algorithm Design:

- Fibonacci Identity:
  $(a^2 + b^2)(c^2 + d^2) = (ac - bd)^2 + (ad + bc)^2$
- Graph Perfect Matching [MVV87]
- Primality Testing [AKS04]

Applications

Various application in Complexity Theory and Algorithm Design:

- Fibonacci Identity:
  $(a^2 + b^2)(c^2 + d^2) = (ac - bd)^2 + (ad + bc)^2$
- Graph Perfect Matching [MVV87]
- Primality Testing [AKS04]
- $\mathrm{IP} = \mathrm{PSPACE}$ [Sha90]

Applications

Various application in Complexity Theory and Algorithm Design:

- Fibonacci Identity:
  $(a^2 + b^2)(c^2 + d^2) = (ac - bd)^2 + (ad + bc)^2$
- Graph Perfect Matching [MVV87]
- Primality Testing [AKS04]
- IP $=$ PSPACE [Sha90]
- PCP Theorem [AS98]

Applications

Various application in Complexity Theory and Algorithm Design:

- Fibonacci Identity:
  $(a^2 + b^2)(c^2 + d^2) = (ac - bd)^2 + (ad + bc)^2$
- Graph Perfect Matching [MVV87]
- Primality Testing [AKS04]
- IP $=$ PSPACE [Sha90]
- PCP Theorem [AS98]
- Geometric Complexity Theory [Mul12]

Applications

Various application in Complexity Theory and Algorithm Design:

- Fibonacci Identity:
  $(a^2+b^2)(c^2+d^2) = (ac-bd)^2 + (ad+bc)^2$
- Graph Perfect Matching [MVV87]
- Primality Testing [AKS04]
- IP $=$ PSPACE [Sha90]
- PCP Theorem [AS98]
- Geometric Complexity Theory [Mul12]
- ...

Existing Algorithms

- Suppose we can generate a random element of a field $\mathbb{F}$. Then we have a general PIT algorithm.

Existing Algorithms

- Suppose we can generate a random element of a field $\mathbb{F}$. Then we have a general PIT algorithm.

Lemma (Schwartz-Zippel)

Let $P \in \mathbb{F}[x_1, \ldots, x_n]$ with total degree bounded by some $d \in \mathbb{N}$. Pick some finite $S \subseteq \mathbb{F}$. Then $\Pr_{\bar{x} \in S^n}[P(\bar{x}) = 0] \leq \frac{d}{|S|}$.

Existing Algorithms

- Suppose we can generate a random element of a field $\mathbb{F}$. Then we have a general PIT algorithm.

Lemma (Schwartz-Zippel)

Let $P \in \mathbb{F}[x_1, \ldots, x_n]$ with total degree bounded by some $d \in \mathbb{N}$. Pick some finite $S \subseteq \mathbb{F}$. Then $\Pr_{\bar{x} \in S^n}[P(\bar{x}) = 0] \leq \frac{d}{|S|}$.

- This does not provide a deterministic algorithm but suggests that one might exist.

Hardness Results

- [KI04]: Deterministic PIT $\Longrightarrow$ super-polynomial circuit lower bounds: Boolean for NEXP or arithmetic for Permanent.

Hardness Results

- [KI04]: Deterministic PIT $\Longrightarrow$ super-polynomial circuit lower bounds: Boolean for NEXP or arithmetic for Permanent.
- [Agr05]: Black-Box PIT $\Longrightarrow$ Explicit exponential lower bounds for general arithmetic formulas.

Hardness Results

- [KI04]: Deterministic PIT $\implies$ super-polynomial circuit lower bounds: Boolean for NEXP or arithmetic for Permanent.
- [Agr05]: Black-Box PIT $\implies$ Explicit exponential lower bounds for general arithmetic formulas.
- [Agr05]: Black-Box PIT for multilinear $\implies$ Explicit exponential lower bounds for multilinear arithmetic formulas.

Hardness Results

- [KI04]: Deterministic PIT $\Longrightarrow$ super-polynomial circuit lower bounds: Boolean for NEXP or arithmetic for Permanent.
- [Agr05]: Black-Box PIT $\Longrightarrow$ Explicit exponential lower bounds for general arithmetic formulas.
- [Agr05]: Black-Box PIT for multilinear $\Longrightarrow$ Explicit exponential lower bounds for multilinear arithmetic formulas.
- [AV08]: Black-Box PIT for depth-4 formulas $\Longrightarrow$ Black-Box PIT for general formulas.

Hardness Results

- [KI04]: Deterministic PIT $\implies$ super-polynomial circuit lower bounds: Boolean for NEXP or arithmetic for Permanent.
- [Agr05]: Black-Box PIT $\implies$ Explicit exponential lower bounds for general arithmetic formulas.
- [Agr05]: Black-Box PIT for multilinear $\implies$ Explicit exponential lower bounds for multilinear arithmetic formulas.
- [AV08]: Black-Box PIT for depth-4 formulas $\implies$ Black-Box PIT for general formulas.

Hardness Results

- [KI04]: Deterministic PIT $\implies$ super-polynomial circuit lower bounds: Boolean for NEXP or arithmetic for Permanent.
- [Agr05]: Black-Box PIT $\implies$ Explicit exponential lower bounds for general arithmetic formulas.
- [Agr05]: Black-Box PIT for multilinear $\implies$ Explicit exponential lower bounds for multilinear arithmetic formulas.
- [AV08]: Black-Box PIT for depth-4 formulas $\implies$ Black-Box PIT for general formulas.

I have a truly marvelous proof of these lower bounds which this slide is too small to contain.

Previous Results

- Sparse Polynomials: [BOT88, KS01, LV03]

Previous Results

- Sparse Polynomials: [BOT88, KS01, LV03]
- Depth-3 formulas: [DS07, KS07, KS09, SS13]

Previous Results

- Sparse Polynomials: [BOT88, KS01, LV03]
- Depth-3 formulas: [DS07, KS07, KS09, SS13]
- Depth-4 formulas: [SV11, ASSS12, KMSV13]

Previous Results

- Sparse Polynomials: [BOT88, KS01, LV03]
- Depth-3 formulas: [DS07, KS07, KS09, SS13]
- Depth-4 formulas: [SV11, ASSS12, KMSV13]
- Bounded-read formulas:
  [ASS12, AvMV15, SV15, FS13, AFS^{+}16]

Multilinear and Read-Once Polynomials

Goal: Find an algorithm that works for certain classes of polynomials.

Multilinear and Read-Once Polynomials

Goal: Find an algorithm that works for certain classes of polynomials.

Definition (Multilinear Polynomial)

Each term of a multilinear polynomial has no variable with degree more than one. For example, $x_1x_2 + x_2x_3 + x_1x_3$.

Multilinear and Read-Once Polynomials

Goal: Find an algorithm that works for certain classes of polynomials.

Definition (Multilinear Polynomial)

Each term of a multilinear polynomial has no variable with degree more than one. For example, $x_1x_2 + x_2x_3 + x_1x_3$.

Lemma (Hitting Set MP)

The set $\{0,1\}^n$ is a hitting set for MPs.

Multilinear and Read-Once Polynomials

Goal: Find an algorithm that works for certain classes of polynomials.

Definition (Multilinear Polynomial)

Each term of a multilinear polynomial has no variable with degree more than one. For example, $x_1x_2 + x_2x_3 + x_1x_3$.

Lemma (Hitting Set MP)

The set $\{0,1\}^n$ is a hitting set for MPs.

Definition (Read-Once Polynomial)

A polynomial that can be expressed by an arithmetic formula f such that no variable appears more than once. For example, $x_1x_2 + x_1x_3$ but not $x_1x_2 + x_2x_3 + x_3x_1$.

Read-Once Results

Lemma (Random Hitting Set)

A random set $\mathcal{H} \subseteq \mathbb{F}^n$ of size n^4 is a hitting set the class of read-once polynomials with high probability.

Read-Once Results

Lemma (Random Hitting Set)

A random set $\mathcal{H} \subseteq \mathbb{F}^n$ of size n^4 is a hitting set the class of read-once polynomials with high probability.

- This suggests that there does exist a hitting set for ROPs.

Read-Once Results

Lemma (Random Hitting Set)

A random set $\mathcal{H} \subseteq \mathbb{F}^n$ of size n^4 is a hitting set the class of read-once polynomials with high probability.

- This suggests that there does exist a hitting set for ROPs.
- This does not suggest how to explicitly find such a set.

Read-Once Results

Lemma (Random Hitting Set)

A random set $\mathcal{H} \subseteq \mathbb{F}^n$ of size n^4 is a hitting set the class of read-once polynomials with high probability.

- This suggests that there does exist a hitting set for ROPs.
- This does not suggest how to explicitly find such a set.

Read-Once Results

Lemma (Random Hitting Set)

A random set $\mathcal{H} \subseteq \mathbb{F}^n$ of size n^4 is a hitting set the class of read-once polynomials with high probability.

- This suggests that there does exist a hitting set for ROPs.
- This does not suggest how to explicitly find such a set.

Theorem ([SV09])

There exists an explicit hitting set for ROPs of size $n^{O(\log n)}$.

Read-Once Results

Lemma (Random Hitting Set)

A random set $\mathcal{H} \subseteq \mathbb{F}^n$ of size n^4 is a hitting set the class of read-once polynomials with high probability.

- This suggests that there does exist a hitting set for ROPs.
- This does not suggest how to explicitly find such a set.

Theorem ([SV09])

There exists an explicit hitting set for ROPs of size $n^{O(\log n)}$.

Theorem (Our Result)

There exists an explicit hitting set of size n^4 for ROPs.

Results and Implications

Theorem (Main: Hitting Set for ROPs)

There exists an explicit hitting set of size n^4 for ROPs.

Results and Implications

Theorem (Main: Hitting Set for ROPs)

There exists an explicit hitting set of size n^4 for ROPs.

Theorem (Corollary 1 from [SV14, BHH95])

There exists a deterministic algorithm that given a black-box access to a ROP outputs a ROF for it in polynomial time.

Results and Implications

Theorem (Main: Hitting Set for ROPs)

There exists an explicit hitting set of size n^4 for ROPs.

Theorem (Corollary 1 from [SV14, BHH95])

There exists a deterministic algorithm that given a black-box access to a ROP outputs a ROF for it in polynomial time.

Theorem (Corollary 2 from [SV15])

For every $k \in \mathbb{N}$ there exists an explicit hitting set of size $n^{O(k)}$ for sums of k ROPs.

Approach

Definition (Generator)

Pick some $k \in \mathbb{N}$ with $k \ll n$. Let $\mathcal{C} \subseteq \mathbb{F}[x_1, \ldots, x_n]$. A *generator* for $\mathcal{C}$ is a polynomial map $G : \mathbb{F}^k \to \mathbb{F}^n$ such that $\forall P \in \mathcal{C}$, we have $P(G) \equiv 0 \iff P \equiv 0$.

Approach

Definition (Generator)

Pick some $k \in \mathbb{N}$ with $k \ll n$. Let $\mathcal{C} \subseteq \mathbb{F}[x_1, \ldots, x_n]$. A *generator* for $\mathcal{C}$ is a polynomial map $G : \mathbb{F}^k \to \mathbb{F}^n$ such that $\forall P \in \mathcal{C}$, we have $P(G) \equiv 0 \iff P \equiv 0$.

Lemma (Hitting Set for General Polynomials)

Let $P \in \mathbb{F}[x_1, \ldots, x_n]$ where the degree of each variable in P is bounded by some $d \in \mathbb{N}$. For any set $S \subseteq \mathbb{F}$ of size at least $d+1$, $\exists \bar{a} \in S^n$ such that $P(\bar{a}) \neq 0$.

Approach

Definition (Generator)

Pick some $k \in \mathbb{N}$ with $k \ll n$. Let $\mathcal{C} \subseteq \mathbb{F}[x_1, \ldots, x_n]$. A *generator* for $\mathcal{C}$ is a polynomial map $G : \mathbb{F}^k \to \mathbb{F}^n$ such that $\forall P \in \mathcal{C}$, we have $P(G) \equiv 0 \iff P \equiv 0$.

Lemma (Hitting Set for General Polynomials)

Let $P \in \mathbb{F}[x_1, \ldots, x_n]$ where the degree of each variable in P is bounded by some $d \in \mathbb{N}$. For any set $S \subseteq \mathbb{F}$ of size at least $d+1$, $\exists \bar{a} \in S^n$ such that $P(\bar{a}) \neq 0$.

Lemma (From Generator to Hitting Set)

Suppose the individual degrees of each component function of G are bounded by some $d \in \mathbb{N}$. Then we can decide if $P(G) \equiv 0$ in time $(nd)^{O(k)}$.

Previous Generator

Definition (The Generator of [SV09])

Pick some $n \in \mathbb{N}$. Pick distinct $\alpha_1, \ldots, \alpha_n \in \mathbb{F}$ and let $\mu_i(w)$ be the Langrange Interpolation Polynomial that evaluates to 0 for α_j with $j \neq i$ and evaluates to 1 at α_i. Then we define:

Previous Generator

Definition (The Generator of [SV09])

Pick some $n \in \mathbb{N}$. Pick distinct $\alpha_1, \ldots, \alpha_n \in \mathbb{F}$ and let $\mu_i(w)$ be the Langrange Interpolation Polynomial that evaluates to 0 for α_j with $j \neq i$ and evaluates to 1 at α_i. Then we define:
$G_{n,1} : \mathbb{F}^2 \rightarrow \mathbb{F}^n$ by $(y, z) \rightarrow (\mu_1(y)z, \ldots, \mu_n(y)z)$.

Previous Generator

Definition (The Generator of [SV09])

Pick some $n \in \mathbb{N}$. Pick distinct $\alpha_1, \ldots, \alpha_n \in \mathbb{F}$ and let $\mu_i(w)$ be the Langrange Interpolation Polynomial that evaluates to 0 for α_j with $j \neq i$ and evaluates to 1 at α_i. Then we define:
$G_{n,1} : \mathbb{F}^2 \rightarrow \mathbb{F}^n$ by $(y, z) \rightarrow (\mu_1(y)z, \ldots, \mu_n(y)z)$.
For $t \in \mathbb{N}$, $G_{n,t} : \mathbb{F}^{2t} \rightarrow \mathbb{F}^n$ as $G_{n,t} = \sum_{i=1}^{t} G_{n,1}(y_i, z_i)$

Previous Generator

Definition (The Generator of [SV09])

Pick some $n \in \mathbb{N}$. Pick distinct $\alpha_1, \ldots, \alpha_n \in \mathbb{F}$ and let $\mu_i(w)$ be the Langrange Interpolation Polynomial that evaluates to 0 for α_j with $j \neq i$ and evaluates to 1 at α_i. Then we define:
$G_{n,1} : \mathbb{F}^2 \to \mathbb{F}^n$ by $(y, z) \to (\mu_1(y)z, \ldots, \mu_n(y)z)$.
For $t \in \mathbb{N}$, $G_{n,t} : \mathbb{F}^{2t} \to \mathbb{F}^n$ as $G_{n,t} = \sum_{i=1}^{t} G_{n,1}(y_i, z_i)$

Theorem ([SV09] - Quasi-Polynomial PIT)

The polynomial map $G_{n,\log n}$ is a generator for ROPs on n variables.

Previous Generator

Definition (The Generator of [SV09])

Pick some $n \in \mathbb{N}$. Pick distinct $\alpha_1, \ldots, \alpha_n \in \mathbb{F}$ and let $\mu_i(w)$ be the Langrange Interpolation Polynomial that evaluates to 0 for α_j with $j \neq i$ and evaluates to 1 at α_i. Then we define:
$G_{n,1} : \mathbb{F}^2 \to \mathbb{F}^n$ by $(y, z) \to (\mu_1(y)z, \ldots, \mu_n(y)z)$.
For $t \in \mathbb{N}$, $G_{n,t} : \mathbb{F}^{2t} \to \mathbb{F}^n$ as $G_{n,t} = \sum_{i=1}^{t} G_{n,1}(y_i, z_i)$

Theorem ([SV09] - Quasi-Polynomial PIT)

The polynomial map $G_{n,\log n}$ is a generator for ROPs on n variables.

Theorem (Our Result - Polynomial PIT)

The polynomial map $G_{n,\mathbf{1}}$ is a generator for ROPs on n variables.

High Level Idea

Definition (Homogenous Polynomial)

A polynomial $P \in \mathbb{F}[x_1, \ldots, x_n]$ is called *homogeneous* if every term in the polynomial has the same total degree.

Lemma (Generator for Homogeneous ROPs)

$G_{n,1}$ is a generator for homogeneous ROPs.

Lemma (From Homogeneous ROPs to General ROPs)

If $P \in \mathbb{F}[x_1, \ldots, x_n]$ is a ROP, then so is $H_{\deg(P)}(P)$.

Corollary (From Homogeneous ROPs to ROPs)

$G_{n,1}$ is a generator for (general) ROPs.

High Level Idea

Definition (Homogenous Polynomial)

A polynomial $P \in \mathbb{F}[x_1, \ldots, x_n]$ is called *homogeneous* if every term in the polynomial has the same total degree.

Lemma (Generator for Homogeneous ROPs)

$G_{n,1}$ is a generator for homogeneous ROPs.

Lemma (From Homogeneous ROPs to General ROPs)

If $P \in \mathbb{F}[x_1, \ldots, x_n]$ is a ROP, then so is $H_{\deg(P)}(P)$.

Corollary (From Homogeneous ROPs to ROPs)

$G_{n,1}$ is a generator for (general) ROPs.

High Level Idea

Definition (Homogenous Polynomial)

A polynomial $P \in \mathbb{F}[x_1, \ldots, x_n]$ is called *homogeneous* if every term in the polynomial has the same total degree.

Lemma (Generator for Homogeneous ROPs)

$G_{n,1}$ is a generator for homogeneous ROPs.

Lemma (From Homogeneous ROPs to General ROPs)

If $P \in \mathbb{F}[x_1, \ldots, x_n]$ is a ROP, then so is $H_{\deg(P)}(P)$.

Corollary (From Homogeneous ROPs to ROPs)

$G_{n,1}$ is a generator for (general) ROPs.

High Level Idea

Definition (Homogenous Polynomial)

A polynomial $P \in \mathbb{F}[x_1, \ldots, x_n]$ is called *homogeneous* if every term in the polynomial has the same total degree.

Lemma (Generator for Homogeneous ROPs)

$G_{n,1}$ is a generator for homogeneous ROPs.

Lemma (From Homogeneous ROPs to General ROPs)

If $P \in \mathbb{F}[x_1, \ldots, x_n]$ is a ROP, then so is $H_{\deg(P)}(P)$.

Corollary (From Homogeneous ROPs to ROPs)

$G_{n,1}$ is a generator for (general) ROPs.

High Level Idea

Definition (Homogenous Polynomial)

A polynomial $P \in \mathbb{F}[x_1, \ldots, x_n]$ is called *homogeneous* if every term in the polynomial has the same total degree.

Lemma (Generator for Homogeneous ROPs)

$G_{n,1}$ is a generator for homogeneous ROPs.

Lemma (From Homogeneous ROPs to General ROPs)

If $P \in \mathbb{F}[x_1, \ldots, x_n]$ is a ROP, then so is $H_{\deg(P)}(P)$.

Corollary (From Homogeneous ROPs to ROPs)

$G_{n,1}$ is a generator for (general) ROPs.

$G_{n,1}$ is a Generator for Homogeneous ROPs: Proof Sketch

$G_{n,1}$ is a Generator for Homogeneous ROPs: Proof Sketch

Pick $\{\alpha_1, \ldots, \alpha_n\} \subseteq \mathbb{F}$. Let $\mu_i(y)$ the i^{th} Lagrange Interpolation Polynomial. Let $G_{n,1}(y, z) = (\mu_1(y)z, \ldots, \mu_n(y)z)$.

$G_{n,1}$ is a Generator for Homogeneous ROPs: Proof Sketch

Pick $\{\alpha_1, \ldots, \alpha_n\} \subseteq \mathbb{F}$. Let $\mu_i(y)$ the i^{th} Lagrange Interpolation Polynomial. Let $G_{n,1}(y, z) = (\mu_1(y)z, \ldots, \mu_n(y)z)$.

Lemma (Homogenous ROP Structural Lemma)

If $P \in \mathbb{F}[x_1, \ldots, x_n]$ is a homogeneous ROP with $n \geq 2$, then $\exists P_1, P_2$ non-constant, variable-disjoint homogeneous ROPs s.t:

1. $P = P_1 \cdot P_2$
2. $P = P_1 + P_2$

$G_{n,1}$ is a Generator for Homogeneous ROPs: Proof Sketch

Pick $\{\alpha_1, \ldots, \alpha_n\} \subseteq \mathbb{F}$. Let $\mu_i(y)$ the i^{th} Lagrange Interpolation Polynomial. Let $G_{n,1}(y, z) = (\mu_1(y)z, \ldots, \mu_n(y)z)$.

Lemma (Homogenous ROP Structural Lemma)

If $P \in \mathbb{F}[x_1, \ldots, x_n]$ is a homogeneous ROP with $n \geq 2$, then $\exists P_1, P_2$ non-constant, variable-disjoint homogeneous ROPs s.t:

1. $P = P_1 \cdot P_2$
2. $P = P_1 + P_2$

By induction on n:

$G_{n,1}$ is a Generator for Homogeneous ROPs: Proof Sketch

Pick $\{\alpha_1, \ldots, \alpha_n\} \subseteq \mathbb{F}$. Let $\mu_i(y)$ the i^{th} Lagrange Interpolation Polynomial. Let $G_{n,1}(y, z) = (\mu_1(y)z, \ldots, \mu_n(y)z)$.

Lemma (Homogenous ROP Structural Lemma)

If $P \in \mathbb{F}[x_1, \ldots, x_n]$ is a homogeneous ROP with $n \geq 2$, then $\exists P_1, P_2$ non-constant, variable-disjoint homogeneous ROPs s.t:

1. $P = P_1 \cdot P_2$
2. $P = P_1 + P_2$

By induction on n:

0. Linear Case (Base case): $P = c_1x_1 + \cdots + c_nx_n$.

$G_{n,1}$ is a Generator for Homogeneous ROPs: Proof Sketch

Pick $\{\alpha_1, \ldots, \alpha_n\} \subseteq \mathbb{F}$. Let $\mu_i(y)$ the i^{th} Lagrange Interpolation Polynomial. Let $G_{n,1}(y, z) = (\mu_1(y)z, \ldots, \mu_n(y)z)$.

Lemma (Homogenous ROP Structural Lemma)

If $P \in \mathbb{F}[x_1, \ldots, x_n]$ is a homogeneous ROP with $n \geq 2$, then $\exists P_1, P_2$ non-constant, variable-disjoint homogeneous ROPs s.t:

1. $P = P_1 \cdot P_2$
2. $P = P_1 + P_2$

By induction on n:

0. Linear Case (Base case): $P = c_1 x_1 + \cdots + c_n x_n$. Pick i such that $c_i \neq 0$. Fix $y = \alpha_i$, then $P(G_{n,1}(\alpha_i, z)) = c_i z$.

$G_{n,1}$ is a Generator for Homogeneous ROPs: Proof Sketch

Pick $\{\alpha_1, \ldots, \alpha_n\} \subseteq \mathbb{F}$. Let $\mu_i(y)$ the i^{th} Lagrange Interpolation Polynomial. Let $G_{n,1}(y, z) = (\mu_1(y)z, \ldots, \mu_n(y)z)$.

Lemma (Homogenous ROP Structural Lemma)

If $P \in \mathbb{F}[x_1, \ldots, x_n]$ is a homogeneous ROP with $n \geq 2$, then $\exists P_1, P_2$ non-constant, variable-disjoint homogeneous ROPs s.t:

1. $P = P_1 \cdot P_2$
2. $P = P_1 + P_2$

By induction on n:

0. Linear Case (Base case): $P = c_1 x_1 + \cdots + c_n x_n$. Pick i such that $c_i \neq 0$. Fix $y = \alpha_i$, then $P(G_{n,1}(\alpha_i, z)) = c_i z$.
1. Multiplicative Case: $P = P_1 \cdot P_2$, where P_i depends on $< n$ variables.

$G_{n,1}$ is a Generator for Homogeneous ROPs: Proof Sketch

Pick $\{\alpha_1, \ldots, \alpha_n\} \subseteq \mathbb{F}$. Let $\mu_i(y)$ the i^{th} Lagrange Interpolation Polynomial. Let $G_{n,1}(y, z) = (\mu_1(y)z, \ldots, \mu_n(y)z)$.

Lemma (Homogenous ROP Structural Lemma)

If $P \in \mathbb{F}[x_1, \ldots, x_n]$ is a homogeneous ROP with $n \geq 2$, then $\exists P_1, P_2$ non-constant, variable-disjoint homogeneous ROPs s.t:

1. $P = P_1 \cdot P_2$
2. $P = P_1 + P_2$

By induction on n:

0. Linear Case (Base case): $P = c_1 x_1 + \cdots + c_n x_n$. Pick i such that $c_i \neq 0$. Fix $y = \alpha_i$, then $P(G_{n,1}(\alpha_i, z)) = c_i z$.
1. Multiplicative Case: $P = P_1 \cdot P_2$, where P_i depends on $< n$ variables. By the inductive hypothesis, $P_1(G_{n,1}), P_2(G_{n,1}) \not\equiv 0$.

$G_{n,1}$ is a Generator for Homogeneous ROPs: Proof Sketch

Pick $\{\alpha_1, \ldots, \alpha_n\} \subseteq \mathbb{F}$. Let $\mu_i(y)$ the i^{th} Lagrange Interpolation Polynomial. Let $G_{n,1}(y, z) = (\mu_1(y)z, \ldots, \mu_n(y)z)$.

Lemma (Homogenous ROP Structural Lemma)

If $P \in \mathbb{F}[x_1, \ldots, x_n]$ is a homogeneous ROP with $n \geq 2$, then $\exists P_1, P_2$ non-constant, variable-disjoint homogeneous ROPs s.t:

1. $P = P_1 \cdot P_2$
2. $P = P_1 + P_2$

By induction on n:

0. Linear Case (Base case): $P = c_1 x_1 + \cdots + c_n x_n$. Pick i such that $c_i \neq 0$. Fix $y = \alpha_i$, then $P(G_{n,1}(\alpha_i, z)) = c_i z$.
1. Multiplicative Case: $P = P_1 \cdot P_2$, where P_i depends on $< n$ variables. By the inductive hypothesis, $P_1(G_{n,1}), P_2(G_{n,1}) \not\equiv 0$. Therefore, $P(G_{n,1}) = P_1(G_{n,1}) \cdot P_2(G_{n,1}) \not\equiv 0$.

$G_{n,1}$ is a Generator for Homogeneous ROPs: Proof Sketch

Pick $\{\alpha_1, \dots, \alpha_n\} \subseteq \mathbb{F}$. Let $\mu_i(y)$ the i^{th} Lagrange Interpolation Polynomial. Let $G_{n,1}(y,z) = (\mu_1(y)z, \dots, \mu_n(y)z)$.

Lemma (Homogenous ROP Structural Lemma)

If $P \in \mathbb{F}[x_1, \dots, x_n]$ is a homogeneous ROP with $n \geq 2$, then $\exists P_1, P_2$ non-constant, variable-disjoint homogeneous ROPs s.t:

1. $P = P_1 \cdot P_2$
2. $P = P_1 + P_2$

By induction on n:

0. Linear Case (Base case): $P = c_1 x_1 + \cdots + c_n x_n$. Pick i such that $c_i \neq 0$. Fix $y = \alpha_i$, then $P(G_{n,1}(\alpha_i, z)) = c_i z$.
1. Multiplicative Case: $P = P_1 \cdot P_2$, where P_i depends on $< n$ variables. By the inductive hypothesis, $P_1(G_{n,1}), P_2(G_{n,1}) \not\equiv 0$. Therefore, $P(G_{n,1}) = P_1(G_{n,1}) \cdot P_2(G_{n,1}) \not\equiv 0$.
2. Additive Case: Main technical contribution of the paper.

Conclusion & Open Questions

Theorem (Black-Box PIT for ROFs)

There exists a polynomial-time black-box PIT algorithm for read-once formulas.

Conclusion & Open Questions

Theorem (Black-Box PIT for ROFs)

There exists a polynomial-time black-box PIT algorithm for read-once formulas.

Theorem (Reconstruction for ROFs)

There exists a polynomial-time reconstruction algorithm for read-once formulas.

Conclusion & Open Questions

Theorem (Black-Box PIT for ROFs)

There exists a polynomial-time black-box PIT algorithm for read-once formulas.

Theorem (Reconstruction for ROFs)

There exists a polynomial-time reconstruction algorithm for read-once formulas.

Open questions:

- Polynomial-time black-box PIT algorithm for read-k formulas?

Conclusion & Open Questions

Theorem (Black-Box PIT for ROFs)

There exists a polynomial-time black-box PIT algorithm for read-once formulas.

Theorem (Reconstruction for ROFs)

There exists a polynomial-time reconstruction algorithm for read-once formulas.

Open questions:

- Polynomial-time black-box PIT algorithm for read-k formulas?
- [AvMV15]: quasi-polynomial-time black-box PIT algorithm for read-k formulas.

Conclusion & Open Questions

Theorem (Black-Box PIT for ROFs)

There exists a polynomial-time black-box PIT algorithm for read-once formulas.

Theorem (Reconstruction for ROFs)

There exists a polynomial-time reconstruction algorithm for read-once formulas.

Open questions:

- Polynomial-time black-box PIT algorithm for read-k formulas?
- [AvMV15]: quasi-polynomial-time black-box PIT algorithm for read-k formulas.
- Our results + [SV15]: Polynomial-time black-box PIT algorithm for sum of two read-once formulas.

Conclusion & Open Questions

Theorem (Black-Box PIT for ROFs)

There exists a polynomial-time black-box PIT algorithm for read-once formulas.

Theorem (Reconstruction for ROFs)

There exists a polynomial-time reconstruction algorithm for read-once formulas.

Open questions:

- Polynomial-time black-box PIT algorithm for read-k formulas?
- [AvMV15]: quasi-polynomial-time black-box PIT algorithm for read-k formulas.
- Our results + [SV15]: Polynomial-time black-box PIT algorithm for sum of two read-once formulas.
- Polynomial-time black-box PIT algorithm for read-twice formulas?

Thank you!

Thank you!

M. Anderson, M. A. Forbes, R. Saptharishi, A. Shpilka, and B. Lee Volk.
Identity testing and lower bounds for read-k oblivious algebraic branching programs.
In *31st Conference on Computational Complexity, CCC*, pages 30:1–30:25, 2016.

M. Agrawal.
Proving lower bounds via pseudo-random generators.
In *Proceedings of the 25th FSTTCS*, volume 3821 of *LNCS*, pages 92–105, 2005.

M. Agrawal, N. Kayal, and N. Saxena.
Primes is in P.
Annals of Mathematics, 160(2):781–793, 2004.

S. Arora and S. Safra.
Probabilistic checking of proofs: A new characterization of NP.

JACM, 45(1):70–122, 1998.

- M. Agrawal, C. Saha, and N. Saxena.
  Quasi-polynomial hitting-set for set-depth-delta formulas.
  2012.
- M. Agrawal, C. Saha, R. Saptharishi, and N. Saxena.
  Jacobian hits circuits: Hitting-sets, lower bounds for depth-d occur-k formulas & depth-3 transcendence degree-k circuits.
  In *Proceedings of the 44th Annual ACM Symposium on Theory of Computing (STOC)*, pages 599–614, 2012.
- M. Agrawal and V. Vinay.
  Arithmetic circuits: A chasm at depth four.
  In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 67–75, 2008.
- M. Anderson, D. van Melkebeek, and I. Volkovich.
  Derandomizing polynomial identity testing for multilinear constant-read formulae.

Computational Complexity, 24(4):695–776, 2015.

N. H. Bshouty, T. R. Hancock, and L. Hellerstein.
Learning arithmetic read-once formulas.
SIAM J. on Computing, 24(4):706–735, 1995.

M. Ben-Or and P. Tiwari.
A deterministic algorithm for sparse multivariate polynominal interpolation.
In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing (STOC)*, pages 301–309, 1988.

Z. Dvir and A. Shpilka.
Locally decodable codes with 2 queries and polynomial identity testing for depth 3 circuits.
SIAM J. on Computing, 36(5):1404–1434, 2007.

M. Forbes and A. Shpilka.
Quasipolynomial-time identity testing of non-commutative and read-once oblivious algebraic branching programs.

In *Proceedings of the 54th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 243–252, 2013.
Full version at http://eccc.hpi-web.de/report/2012/115.

V. Kabanets and R. Impagliazzo.
Derandomizing polynomial identity tests means proving circuit lower bounds.
Computational Complexity, 13(1-2):1–46, 2004.

Z. S. Karnin, P. Mukhopadhyay, A. Shpilka, and I. Volkovich.
Deterministic identity testing of depth 4 multilinear circuits with bounded top fan-in.
SIAM J. on Computing, 42(6):2114–2131, 2013.

A. Klivans and D. Spielman.
Randomness efficient identity testing of multivariate polynomials.

In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 216–223, 2001.

N. Kayal and N. Saxena.
Polynomial identity testing for depth 3 circuits.
Computational Complexity, 16(2):115–138, 2007.

N. Kayal and S. Saraf.
Blackbox polynomial identity testing for depth 3 circuits.
In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 198–207, 2009.
Full version at http://eccc.hpi-web.de/report/2009/032.

R. J. Lipton and N. K. Vishnoi.
Deterministic identity testing for multivariate polynomials.
In *Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 756–760, 2003.

K. Mulmuley.

Geometric complexity theory v: Equivalence between blackbox derandomization of polynomial identity testing and derandomization of noether's normalization lemma.
In *Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 629–638, 2012.

K. Mulmuley, U. Vazirani, and V. Vazirani.
Matching is as easy as matrix inversion.
Combinatorica, 7(1):105–113, 1987.

A. Shamir.
IP=PSPACE.
In *Proceedings of the Thirty First Annual Symposium on Foundations of Computer Science*, pages 11–15, 1990.

N. Saxena and C. Seshadhri.
From sylvester-gallai configurations to rank bounds: Improved blackbox identity test for depth-3 circuits.

J. ACM, 60(5):33, 2013.

A. Shpilka and I. Volkovich.
Improved polynomial identity testing for read-once formulas.
In *APPROX-RANDOM*, pages 700–713, 2009.
Full version at http://eccc.hpi-web.de/report/2010/011.

S. Saraf and I. Volkovich.
Blackbox identity testing for depth-4 multilinear circuits.
In *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing (STOC)*, pages 421–430, 2011.
Full version at http://eccc.hpi-web.de/report/2011/046.

A. Shpilka and I. Volkovich.
On reconstruction and testing of read-once formulas.
Theory of Computing, 10:465–514, 2014.

A. Shpilka and I. Volkovich.
Read-once polynomial identity testing.
Computational Complexity, 24(3):477–532, 2015.