Hardness and Independence of Polynomials

Prerona Chatterjee
Tata Institute of Fundamental Research, Mumbai

March 17, 2022

Introduction

Polynomials are important!

Polynomials are important!

Algebraic Circuit Complexity

Polynomials are important!

Algebraic Circuit Complexity

How efficiently can a given computational model compute the given polynomial?

Introduction

Polynomials are important!

Algebraic Circuit Complexity

How efficiently can a given computational model compute the given polynomial?

Identity Testing and Algebraic Independence

Polynomials are important!

Algebraic Circuit Complexity

How efficiently can a given computational model compute the given polynomial?

Identity Testing and Algebraic Independence

Given a set of polynomials, how efficiently can one check whether they are algebraically independent?

Introduction

Polynomials are important!

Algebraic Circuit Complexity

How efficiently can a given computational model compute the given polynomial?

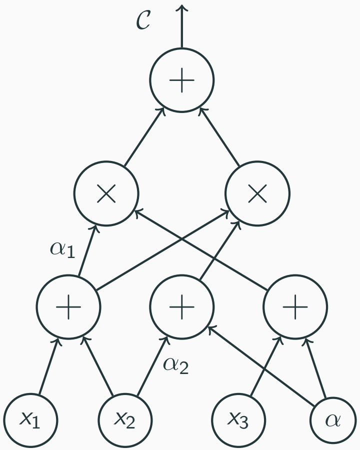
Identity Testing and Algebraic Independence

Given a set of polynomials, how efficiently can one check whether they are algebraically independent?

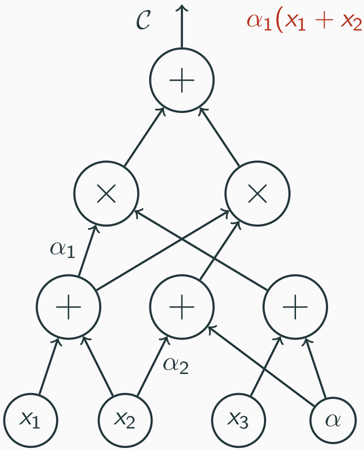
Given an algebraic circuit as input, how efficiently can one check if it computes the identically zero polynommial?

Part 1: Lower Bounds in Algebraic Circuit Complexity

Algebraic Models of Computation

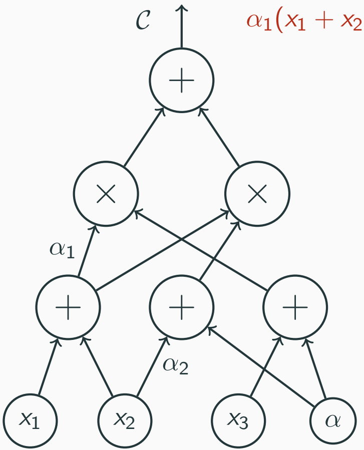


Algebraic Models of Computation

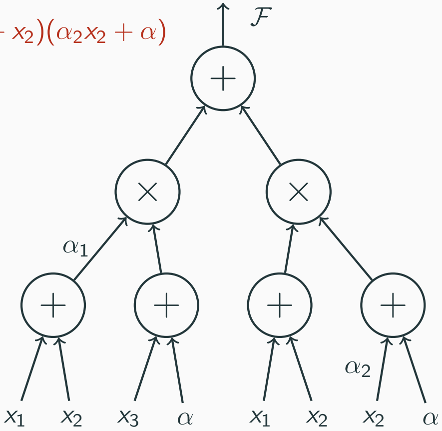


$\alpha_1(x_1 + x_2)(x_3 + \alpha) + (x_1 + x_2)(\alpha_2 x_2 + \alpha)$

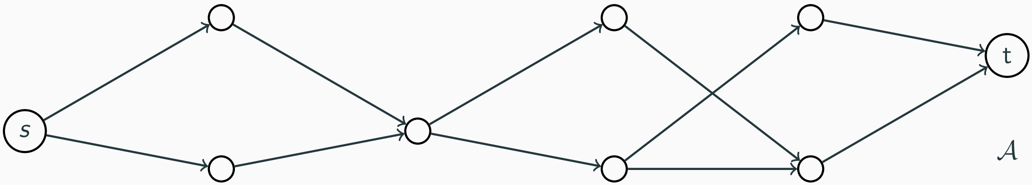
Algebraic Models of Computation



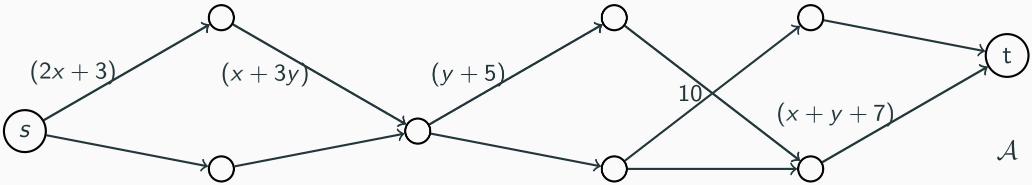
$$\alpha_1(x_1 + x_2)(x_3 + \alpha) + (x_1 + x_2)(\alpha_2 x_2 + \alpha)$$



Algebraic Branching Programs

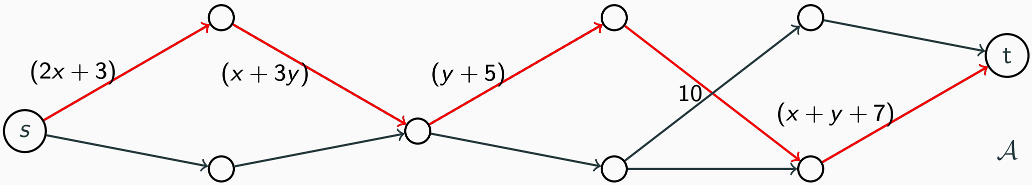


Algebraic Branching Programs



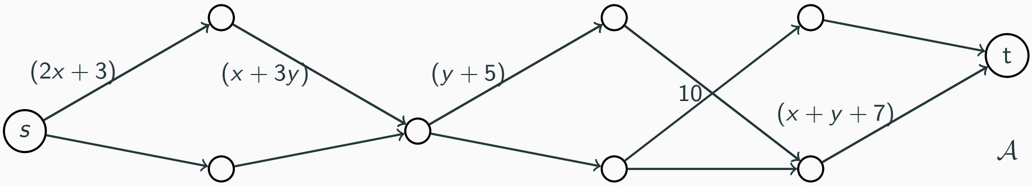
- Label on each edge: An affine linear form in $\{x_1, x_2, \ldots, x_n\}$

Algebraic Branching Programs



- Label on each edge: An affine linear form in $\{x_1, x_2, \ldots, x_n\}$
- Polynomial computed by the path $p = \mathrm{wt}(p)$: Product of the edge labels on p

Algebraic Branching Programs



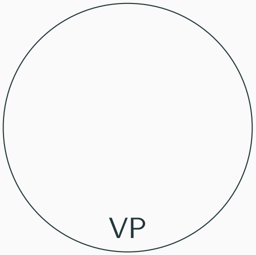
- Label on each edge: An affine linear form in $\{x_1, x_2, \ldots, x_n\}$
- Polynomial computed by the path $p = \mathrm{wt}(p)$: Product of the edge labels on p
- Polynomial computed by the ABP: $f_{\mathcal{A}}(\mathbf{x}) = \sum_p \mathrm{wt}(p)$

Lower Bounds in Algebraic Circuit Complexity

Objects of Study: Polynomials over n variables of degree d.

Objects of Study: Polynomials over n variables of degree d.

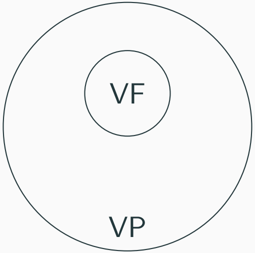
VP: Polynomials computable by circuits of size $\text{poly}(n, d)$.



Objects of Study: Polynomials over n variables of degree d.

VF: Polynomials computable by formulas of size $\text{poly}(n, d)$.

VP: Polynomials computable by circuits of size $\text{poly}(n, d)$.



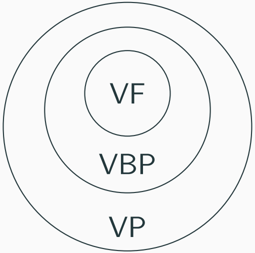
Lower Bounds in Algebraic Circuit Complexity

Objects of Study: Polynomials over n variables of degree d.

VF: Polynomials computable by formulas of size $\mathrm{poly}(n, d)$.

VBP: Polynomials computable by ABPs of size $\mathrm{poly}(n, d)$.

VP: Polynomials computable by circuits of size $\mathrm{poly}(n, d)$.

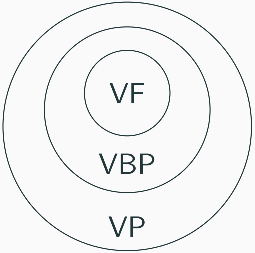


Objects of Study: Polynomials over n variables of degree d.

VF: Polynomials computable by formulas of size poly(n, d).

VBP: Polynomials computable by ABPs of size poly(n, d).

VP: Polynomials computable by circuits of size poly(n, d).



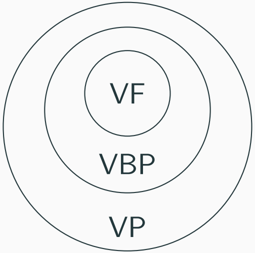
Are the inclusions tight?

Objects of Study: Polynomials over n variables of degree d.

VF: Polynomials computable by formulas of size $\text{poly}(n, d)$.

VBP: Polynomials computable by ABPs of size $\text{poly}(n, d)$.

VP: Polynomials computable by circuits of size $\text{poly}(n, d)$.



Are the inclusions tight?

Central Question: Find explicit polynomials that cannot be computed by efficient circuits.

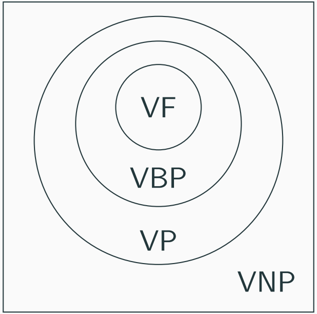
Objects of Study: Polynomials over n variables of degree d.

VF: Polynomials computable by formulas of size $\text{poly}(n, d)$.

VBP: Polynomials computable by ABPs of size $\text{poly}(n, d)$.

VP: Polynomials computable by circuits of size $\text{poly}(n, d)$.

VNP: Explicit Polynomials



Are the inclusions tight?

Central Question: Find explicit polynomials that cannot be computed by efficient circuits.

General Circuits

[Baur-Strassen]: Any algebraic circuit computing $\sum_{i=1}^{n} x_i^d$ requires $\Omega(n \log d)$ wires.

Lower Bounds for General Models

General Circuits

[Baur-Strassen]: Any algebraic circuit computing $\sum_{i=1}^{n} x_i^d$ requires $\Omega(n \log d)$ wires.

Homogeneous ABPs

[Kumar]: Any ABP with $(d+1)$ layers computing $\sum_{i=1}^{n} x_i^d$ requires $\Omega(nd)$ vertices.

Lower Bounds for General Models

General Circuits

[Baur-Strassen]: Any algebraic circuit computing $\sum_{i=1}^{n} x_i^d$ requires $\Omega(n \log d)$ wires.

Homogeneous ABPs

[Kumar]: Any ABP with $(d+1)$ layers computing $\sum_{i=1}^{n} x_i^d$ requires $\Omega(nd)$ vertices.

General ABPs

[C-Kumar-She-Volk]: Any ABP computing $\sum_{i=1}^{n} x_i^d$ requires $\Omega(nd)$ vertices.

Some Remarks

1. If the edges are allowed to be polynomials of degree $\leq \Delta$, we get an $\Omega(nd/\Delta)$ lower bound.

Some Remarks

1. If the edges are allowed to be polynomials of degree $\leq \Delta$, we get an $\Omega(nd/\Delta)$ lower bound.
2. In the unlayered case, if the edges are labelled by polynomials of degree at most Δ, the lower bound we get is $\Omega(n \log n/\Delta \log \log n)$.

Some Remarks

1. If the edges are allowed to be polynomials of degree $\leq \Delta$, we get an $\Omega(nd/\Delta)$ lower bound.
2. In the unlayered case, if the edges are labelled by polynomials of degree at most Δ, the lower bound we get is $\Omega(n \log n/\Delta \log\log n)$.
3. The lower bound is also true for a multilinear polynomial

$$\mathrm{ESYM}_{n,0.1n}(\mathbf{x}) = \sum_{i_1 < \cdots < i_{0.1n} \in [n]} \sum_{j=1}^{n} x_{i_j}.$$

Lower Bounds for General Formulas

[Kalorkoti]: Any formula computing the n^2-variate $\mathrm{Det}_n(\mathbf{x})$ requires $\Omega(n^3)$ vertices.

Lower Bounds for General Formulas

[Kalorkoti]: Any formula computing the n^2-variate $\mathrm{Det}_n(\mathbf{x})$ requires $\Omega(n^3)$ vertices.

[Shpilka-Yehudayoff] (using Kalorkoti's method): There is an n-variate multilinear polynomial such that any formula computing it requires $\Omega(n^2/\log n)$ vertices.

Lower Bounds for General Formulas

[Kalorkoti]: Any formula computing the n^2-variate $\mathrm{Det}_n(\mathbf{x})$ requires $\Omega(n^3)$ vertices.

[Shpilka-Yehudayoff] (using Kalorkoti's method): There is an n-variate multilinear polynomial such that any formula computing it requires $\Omega(n^2/\log n)$ vertices.

[C-Kumar-She-Volk]: Any formula computing $\mathsf{ESym}_{n,0.1n}(\mathbf{x})$ requires $\Omega(n^2)$ vertices, where

$$\mathrm{ESYM}_{n,d}(\mathbf{x}) = \sum_{i_1 < \cdots < i_d \in [n]} x_{i_1} \cdots x_{i_d}.$$

Some Remarks

1. • SY had shown that any formula computing $\sum_{i=1}^{n}\sum_{j=1}^{n} x_i^j y_j$ requires $\Omega(n^2)$ wires.

Some Remarks

1.
- SY had shown that any formula computing $\sum_{i=1}^{n}\sum_{j=1}^{n} x_i^j y_j$ requires $\Omega(n^2)$ wires.
- But it is trivial to show an *nd* lower bound for polynomials over *n* variables that have *individual degree* at least *d*.

Some Remarks

1.
    - SY had shown that any formula computing $\sum_{i=1}^{n}\sum_{j=1}^{n} x_i^j y_j$ requires $\Omega(n^2)$ wires.
    - But it is trivial to show an *nd* lower bound for polynomials over *n* variables that have *individual degree* at least d.
    - Multilinearising the SY polynomial gives an $\Omega(n^2/\log n)$ lower bound.

Some Remarks

1. - SY had shown that any formula computing $\sum_{i=1}^{n}\sum_{j=1}^{n} x_i^j y_j$ requires $\Omega(n^2)$ wires.
   - But it is trivial to show an nd lower bound for polynomials over n variables that have *individual degree* at least d.
   - Multilinearising the SY polynomial gives an $\Omega(n^2/\log n)$ lower bound.
2. Kalorkoti's method can not give a better bound against multilinear polynomials [Jukna].

Some Remarks

1. - SY had shown that any formula computing $\sum_{i=1}^{n} \sum_{j=1}^{n} x_i^j y_j$ requires $\Omega(n^2)$ wires.
   - But it is trivial to show an nd lower bound for polynomials over n variables that have *individual degree* at least d.
   - Multilinearising the SY polynomial gives an $\Omega(n^2/\log n)$ lower bound.
2. Kalorkoti's method can not give a better bound against multilinear polynomials [Jukna].
3. Our result also shows a super-linear separation between the computational powers of circuits and formulas when computing multilinear polynomials.

Proof Overview: The ABP Lower Bound

Step 0 ([Kumar]): Look at the homogeneous case

Any ABP with $(d+1)$ layers computing $\sum_{i=1}^{n} x_i^d$ has $\Omega(nd)$ vertices.

Proof Overview: The ABP Lower Bound

Step 0 ([Kumar]): Look at the homogeneous case

Any ABP with $(d+1)$ layers computing $\sum_{i=1}^{n} x_i^d$ has $\Omega(nd)$ vertices.

Step 1: Generalise above statement to get the base case

Any ABP with $(d+1)$ layers

Proof Overview: The ABP Lower Bound

Step 0 ([Kumar]): Look at the homogeneous case

Any ABP with $(d+1)$ layers computing $\sum_{i=1}^{n} x_i^d$ has $\Omega(nd)$ vertices.

Step 1: Generalise above statement to get the base case

Any ABP with $(d+1)$ layers computing a polynomial of the form

$$f = \sum_{i=1}^{n} x_i^d + \sum_{i=1}^{r} A_i(\mathbf{x}) \cdot B_i(\mathbf{x}) + \delta(\mathbf{x})$$

Proof Overview: The ABP Lower Bound

Step 0 ([Kumar]): Look at the homogeneous case

Any ABP with $(d+1)$ layers computing $\sum_{i=1}^{n} x_i^d$ has $\Omega(nd)$ vertices.

Step 1: Generalise above statement to get the base case

Any ABP with $(d+1)$ layers computing a polynomial of the form

$$f = \sum_{i=1}^{n} x_i^d + \sum_{i=1}^{r} A_i(\mathbf{x}) \cdot B_i(\mathbf{x}) + \delta(\mathbf{x})$$

where $A_i(0) = 0 = B_i(0)$ and $\deg(\delta(\mathbf{x})) < d$,

Proof Overview: The ABP Lower Bound

Step 0 ([Kumar]): Look at the homogeneous case

Any ABP with $(d+1)$ layers computing $\sum_{i=1}^{n} x_i^d$ has $\Omega(nd)$ vertices.

Step 1: Generalise above statement to get the base case

Any ABP with $(d+1)$ layers computing a polynomial of the form

$$f = \sum_{i=1}^{n} x_i^d + \sum_{i=1}^{r} A_i(\mathbf{x}) \cdot B_i(\mathbf{x}) + \delta(\mathbf{x})$$

where $A_i(0) = 0 = B_i(0)$ and $\deg(\delta(\mathbf{x})) < d$, has at least

$((n/2) - r) \cdot (d - 1)$ vertices.

Proof Overview: The ABP Lower Bound

Step 2: Iteratively reduce to Base Case

In each iteration, reduce the number of layers till it becomes $(d+1)$ such that

Proof Overview: The ABP Lower Bound

Step 2: Iteratively reduce to Base Case

In each iteration, reduce the number of layers till it becomes $(d+1)$ such that

- the number of layers is reduced by a constant fraction,

Proof Overview: The ABP Lower Bound

Step 2: Iteratively reduce to Base Case

In each iteration, reduce the number of layers till it becomes $(d + 1)$ such that

- the number of layers is reduced by a constant fraction,
- the size does not increase,

Proof Overview: The ABP Lower Bound

Step 2: Iteratively reduce to Base Case

In each iteration, reduce the number of layers till it becomes $(d+1)$ such that

- the number of layers is reduced by a constant fraction,
- the size does not increase,
- the polynomial being computed continues to look like

$$f_{\ell+1} = \sum_{i=1}^{n} x_i^d + \sum_{i=1}^{r_{\ell+1}} A_i(\mathbf{x}) \cdot B_i(\mathbf{x}) + \delta_{\ell+1}(\mathbf{x})$$

where $A_i(0) = 0 = B_i(0)$ and $\deg(\delta_{\ell+1}(\mathbf{x})) < d$,

Proof Overview: The ABP Lower Bound

Step 2: Iteratively reduce to Base Case

In each iteration, reduce the number of layers till it becomes $(d+1)$ such that

- the number of layers is reduced by a constant fraction,
- the size does not increase,
- the polynomial being computed continues to look like

$$f_{\ell+1} = \sum_{i=1}^{n} x_i^d + \sum_{i=1}^{r_{\ell+1}} A_i(\mathbf{x}) \cdot B_i(\mathbf{x}) + \delta_{\ell+1}(\mathbf{x})$$

  where $\quad A_i(0) = 0 = B_i(0) \quad$ and $\quad \deg(\delta_{\ell+1}(\mathbf{x})) < d$,

- number of error terms collected is small.

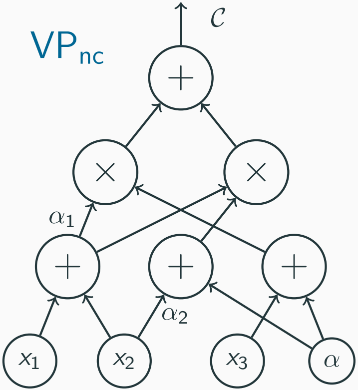
Questions?

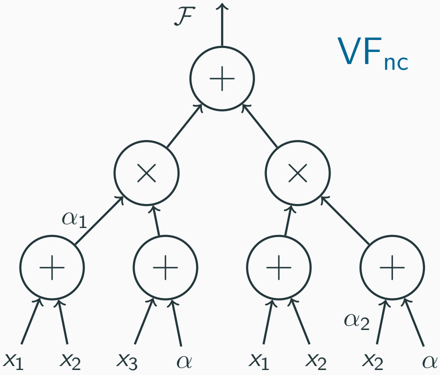
The Non-Commutative Setting

$$f(x, y) = (x + y)^2 = (x + y) \times (x + y) = x^2 + xy + yx + y^2 \neq x^2 + 2xy + y^2$$

The Non-Commutative Setting

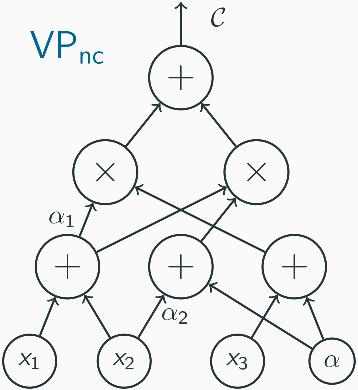
$$f(x,y) = (x+y)^2 = (x+y)\times(x+y) = x^2 + xy + yx + y^2 \neq x^2 + 2xy + y^2$$



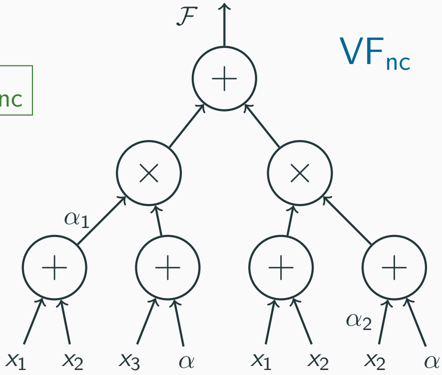


The Non-Commutative Setting

$$f(x, y) = (x + y)^2 = (x + y) \times (x + y) = x^2 + xy + yx + y^2 \neq x^2 + 2xy + y^2$$



$$\mathsf{VP}_{\mathsf{nc}} \supseteq \mathsf{VF}_{\mathsf{nc}}$$



Lower Bounds for General Non-Commutative Models

Is there an explicit polynomial that is outside $\mathsf{VP}_{\mathsf{nc}}$?

Lower Bounds for General Non-Commutative Models

Is there an explicit polynomial that is outside $\mathsf{VP}_{\mathsf{nc}}$?

Circuits [Baur - Strassen]: $\Omega(n \log d)$ for an n-variate, degree d polynomial.

Lower Bounds for General Non-Commutative Models

Is there an explicit polynomial that is outside $\mathsf{VP}_{\mathsf{nc}}$? What about $\mathsf{VF}_{\mathsf{nc}}$?

Circuits [Baur - Strassen]: $\Omega(n \log d)$ for an n-variate, degree d polynomial.

Lower Bounds for General Non-Commutative Models

Is there an explicit polynomial that is outside $\mathsf{VP}_{\mathsf{nc}}$? What about $\mathsf{VF}_{\mathsf{nc}}$?

Circuits [Baur - Strassen]: $\Omega(n \log d)$ for an n-variate, degree d polynomial.

Formulas [Nisan]: $2^{\Omega(n)}$ for a 2-variate, degree n polynomial in $\mathsf{VP}_{\mathsf{nc}}$.

Lower Bounds for General Non-Commutative Models

Is there an explicit polynomial that is outside $\mathsf{VP}_{\mathsf{nc}}$? What about $\mathsf{VF}_{\mathsf{nc}}$?

Circuits [Baur - Strassen]: $\Omega(n \log d)$ for an n-variate, degree d polynomial.

Formulas [Nisan]: $2^{\Omega(n)}$ for a 2-variate, degree n polynomial in $\mathsf{VP}_{\mathsf{nc}}$. So, $\mathsf{VF}_{\mathsf{nc}} \neq \mathsf{VP}_{\mathsf{nc}}$.

Lower Bounds for General Non-Commutative Models

Is there an explicit polynomial that is outside $\mathsf{VP}_{\mathsf{nc}}$? What about $\mathsf{VF}_{\mathsf{nc}}$?

Circuits [Baur - Strassen]: $\Omega(n \log d)$ for an n-variate, degree d polynomial.

Formulas [Nisan]: $2^{\Omega(n)}$ for a 2-variate, degree n polynomial in $\mathsf{VP}_{\mathsf{nc}}$. So, $\mathsf{VF}_{\mathsf{nc}} \neq \mathsf{VP}_{\mathsf{nc}}$.

But the proof is via a lower bound against non-commutative Algebraic Branching Programs.

Lower Bounds for General Non-Commutative Models

Is there an explicit polynomial that is outside $\mathsf{VP}_{\mathsf{nc}}$? What about $\mathsf{VF}_{\mathsf{nc}}$?

Circuits [Baur - Strassen]: $\Omega(n \log d)$ for an n-variate, degree d polynomial.

Formulas [Nisan]: $2^{\Omega(n)}$ for a 2-variate, degree n polynomial in $\mathsf{VP}_{\mathsf{nc}}$. So, $\mathsf{VF}_{\mathsf{nc}} \neq \mathsf{VP}_{\mathsf{nc}}$.

But the proof is via a lower bound against non-commutative Algebraic Branching Programs.

$\mathsf{VBP}_{\mathsf{nc}}$

Lower Bounds for General Non-Commutative Models

Is there an explicit polynomial that is outside $\mathsf{VP_{nc}}$? What about $\mathsf{VF_{nc}}$?

Circuits [Baur - Strassen]: $\Omega(n \log d)$ for an n-variate, degree d polynomial.

Formulas [Nisan]: $2^{\Omega(n)}$ for a 2-variate, degree n polynomial in $\mathsf{VP_{nc}}$. So, $\mathsf{VF_{nc}} \neq \mathsf{VP_{nc}}$.

But the proof is via a lower bound against non-commutative Algebraic Branching Programs.

$$\mathsf{VF_{nc}} \subseteq \mathsf{VBP_{nc}} \subseteq \mathsf{VP_{nc}}$$

Lower Bounds for General Non-Commutative Models

Is there an explicit polynomial that is outside $\mathsf{VP}_{\mathsf{nc}}$? What about $\mathsf{VF}_{\mathsf{nc}}$?

Circuits [Baur - Strassen]: $\Omega(n \log d)$ for an n-variate, degree d polynomial.

Formulas [Nisan]: $2^{\Omega(n)}$ for a 2-variate, degree n polynomial in $\mathsf{VP}_{\mathsf{nc}}$. So, $\mathsf{VF}_{\mathsf{nc}} \neq \mathsf{VP}_{\mathsf{nc}}$.

But the proof is via a lower bound against non-commutative Algebraic Branching Programs.

$$\mathsf{VF}_{\mathsf{nc}} \subseteq \mathsf{VBP}_{\mathsf{nc}} \subseteq \mathsf{VP}_{\mathsf{nc}}$$

So Nisan actually showed that $\mathsf{VBP}_{\mathsf{nc}} \neq \mathsf{VP}_{\mathsf{nc}}$.

The ABP vs Formula Question

The Question [Nisan]: Is $\mathsf{VF}_{\mathsf{nc}} = \mathsf{VBP}_{\mathsf{nc}}$?

The ABP vs Formula Question

The Question [Nisan]: Is $\mathsf{VF}_{\mathsf{nc}} = \mathsf{VBP}_{\mathsf{nc}}$?

Note: Every monomial in a non-commutative polynomial $f(x_1, \ldots, x_n)$ can be thought of as a word over the underlying variables $\{x_1, \ldots, x_n\}$.

The ABP vs Formula Question

The Question [Nisan]: Is $\mathrm{VF}_{\mathrm{nc}} = \mathrm{VBP}_{\mathrm{nc}}$?

Note: Every monomial in a non-commutative polynomial $f(x_1, \ldots, x_n)$ can be thought of as a word over the underlying variables $\{x_1, \ldots, x_n\}$.

Definitions

The ABP vs Formula Question

The Question [Nisan]: Is $\mathsf{VF}_{\mathsf{nc}} = \mathsf{VBP}_{\mathsf{nc}}$?

Note: Every monomial in a non-commutative polynomial $f(x_1, \ldots, x_n)$ can be thought of as a word over the underlying variables $\{x_1, \ldots, x_n\}$.

Definitions Let $\{X_1, \ldots, X_m\}$ be a partition of the variables into buckets.

The ABP vs Formula Question

The Question [Nisan]: Is $\mathrm{VF}_{\mathrm{nc}} = \mathrm{VBP}_{\mathrm{nc}}$?

Note: Every monomial in a non-commutative polynomial $f(x_1, \ldots, x_n)$ can be thought of as a word over the underlying variables $\{x_1, \ldots, x_n\}$.

Definitions Let $\{X_1, \ldots, X_m\}$ be a partition of the variables into buckets.

Abecedarian Polynomials: Polynomials in which every monomial has the form $X_1^* X_2^* \cdots X_m^*$.

The ABP vs Formula Question

The Question [Nisan]: Is $\mathrm{VF_{nc}} = \mathrm{VBP_{nc}}$?

Note: Every monomial in a non-commutative polynomial $f(x_1, \ldots, x_n)$ can be thought of as a word over the underlying variables $\{x_1, \ldots, x_n\}$.

Definitions Let $\{X_1, \ldots, X_m\}$ be a partition of the variables into buckets.

Abecedarian Polynomials: Polynomials in which every monomial has the form $X_1^* X_2^* \cdots X_m^*$.

X_1 X_2

The ABP vs Formula Question

The Question [Nisan]: Is $\mathsf{VF}_{\mathsf{nc}} = \mathsf{VBP}_{\mathsf{nc}}$?

Note: Every monomial in a non-commutative polynomial $f(x_1, \ldots, x_n)$ can be thought of as a word over the underlying variables $\{x_1, \ldots, x_n\}$.

Definitions Let $\{X_1, \ldots, X_m\}$ be a partition of the variables into buckets.

Abecedarian Polynomials: Polynomials in which every monomial has the form $X_1^* X_2^* \cdots X_m^*$.

$$X_1 = \{x_1\} \qquad X_2 = \{x_2\}$$

The ABP vs Formula Question

The Question [Nisan]: Is $\mathrm{VF}_{\mathrm{nc}} = \mathrm{VBP}_{\mathrm{nc}}$?

Note: Every monomial in a non-commutative polynomial $f(x_1, \ldots, x_n)$ can be thought of as a word over the underlying variables $\{x_1, \ldots, x_n\}$.

Definitions Let $\{X_1, \ldots, X_m\}$ be a partition of the variables into buckets.

Abecedarian Polynomials: Polynomials in which every monomial has the form $X_1^* X_2^* \cdots X_m^*$.

$X_1 = \{x_1\}$ $X_2 = \{x_2\}$ $x_1 x_1 x_2$

The ABP vs Formula Question

The Question [Nisan]: Is $\mathrm{VF}_{\mathrm{nc}} = \mathrm{VBP}_{\mathrm{nc}}$?

Note: Every monomial in a non-commutative polynomial $f(x_1, \ldots, x_n)$ can be thought of as a word over the underlying variables $\{x_1, \ldots, x_n\}$.

Definitions Let $\{X_1, \ldots, X_m\}$ be a partition of the variables into buckets.

Abecedarian Polynomials: Polynomials in which every monomial has the form $X_1^* X_2^* \cdots X_m^*$.

$X_1 = \{x_1\}$ $X_2 = \{x_2\}$ $x_1 x_1 x_2$ x_2

The ABP vs Formula Question

The Question [Nisan]: Is $\mathrm{VF}_{\mathrm{nc}} = \mathrm{VBP}_{\mathrm{nc}}$?

Note: Every monomial in a non-commutative polynomial $f(x_1, \ldots, x_n)$ can be thought of as a word over the underlying variables $\{x_1, \ldots, x_n\}$.

Definitions Let $\{X_1, \ldots, X_m\}$ be a partition of the variables into buckets.

Abecedarian Polynomials: Polynomials in which every monomial has the form $X_1^* X_2^* \cdots X_m^*$.

$X_1 = \{x_1\}$ $X_2 = \{x_2\}$ $x_1 x_1 x_2$ x_2 $x_1 x_2 x_1$

The ABP vs Formula Question

The Question [Nisan]: Is $\mathsf{VF}_{\mathsf{nc}} = \mathsf{VBP}_{\mathsf{nc}}$?

Note: Every monomial in a non-commutative polynomial $f(x_1, \ldots, x_n)$ can be thought of as a word over the underlying variables $\{x_1, \ldots, x_n\}$.

Definitions Let $\{X_1, \ldots, X_m\}$ be a partition of the variables into buckets.

Abecedarian Polynomials: Polynomials in which every monomial has the form $X_1^* X_2^* \cdots X_m^*$.

The ABP vs Formula Question

The Question [Nisan]: Is $\mathrm{VF}_{\mathrm{nc}} = \mathrm{VBP}_{\mathrm{nc}}$?

Note: Every monomial in a non-commutative polynomial $f(x_1, \ldots, x_n)$ can be thought of as a word over the underlying variables $\{x_1, \ldots, x_n\}$.

Definitions Let $\{X_1, \ldots, X_m\}$ be a partition of the variables into buckets.

Abecedarian Polynomials: Polynomials in which every monomial has the form $X_1^* X_2^* \cdots X_m^*$.

Syntactically Abecedarian Formulas: Non-commutative formulas with a syntactic restriction that makes them naturally compute abecedarian polynomials.

The ABP vs Formula Question

The Question [Nisan]: Is $\mathrm{VF}_{\mathrm{nc}} = \mathrm{VBP}_{\mathrm{nc}}$?

Note: Every monomial in a non-commutative polynomial $f(x_1, \ldots, x_n)$ can be thought of as a word over the underlying variables $\{x_1, \ldots, x_n\}$.

Definitions Let $\{X_1, \ldots, X_m\}$ be a partition of the variables into buckets.

Abecedarian Polynomials: Polynomials in which every monomial has the form $X_1^* X_2^* \cdots X_m^*$.

Syntactically Abecedarian Formulas: Non-commutative formulas with a syntactic restriction that makes them naturally compute abecedarian polynomials.

Main Result:

There is a tight superpolynomial separation between *abecedarian* formulas and ABPs.

Proof Idea

The Statement: There is an explicit n^2-variate, degree-d abcd-polynomial $f_{n,d}(\mathbf{x})$ such that

Proof Idea

The Statement: There is an explicit n^2-variate, degree-d abcd-polynomial $f_{n,d}(\mathbf{x})$ such that

- abcd-ABP Upper Bound: the abcd-ABP complexity of $f_{n,d}(\mathbf{x})$ is $\Theta(nd)$;

Proof Idea

The Statement: There is an explicit n^2-variate, degree-d abcd-polynomial $f_{n,d}(\mathbf{x})$ such that

- abcd-ABP Upper Bound: the abcd-ABP complexity of $f_{n,d}(\mathbf{x})$ is $\Theta(nd)$;
- abcd-Formula Lower Bound: the abcd-formula complexity of $f_{n,\log n}(\mathbf{x})$ is $n^{\Theta(\log\log n)}$.

Proof Idea

The Statement: There is an explicit n^2-variate, degree-d abcd-polynomial $f_{n,d}(\mathbf{x})$ such that

- abcd-ABP Upper Bound: the abcd-ABP complexity of $f_{n,d}(\mathbf{x})$ is $\Theta(nd)$;
- abcd-Formula Lower Bound: the abcd-formula complexity of $f_{n,\log n}(\mathbf{x})$ is $n^{\Theta(\log \log n)}$.

The Proof Idea:

Proof Idea

The Statement: There is an explicit n^2-variate, degree-d abcd-polynomial $f_{n,d}(\mathbf{x})$ such that

- abcd-ABP Upper Bound: the abcd-ABP complexity of $f_{n,d}(\mathbf{x})$ is $\Theta(nd)$;
- abcd-Formula Lower Bound: the abcd-formula complexity of $f_{n,\log n}(\mathbf{x})$ is $n^{\Theta(\log \log n)}$.

The Proof Idea:

4. Use known lower bound against homogeneous multilinear formulas [HY11].

Proof Idea

The Statement: There is an explicit n^2-variate, degree-d abcd-polynomial $f_{n,d}(\mathbf{x})$ such that

- abcd-ABP Upper Bound: the abcd-ABP complexity of $f_{n,d}(\mathbf{x})$ is $\Theta(nd)$;
- abcd-Formula Lower Bound: the abcd-formula complexity of $f_{n,\log n}(\mathbf{x})$ is $n^{\Theta(\log \log n)}$.

The Proof Idea:

1. Use low degree to make the abcd-formula structured.

4. Use known lower bound against homogeneous multilinear formulas [HY11].

Proof Idea

The Statement: There is an explicit n^2-variate, degree-d abcd-polynomial $f_{n,d}(\mathbf{x})$ such that

- abcd-ABP Upper Bound: the abcd-ABP complexity of $f_{n,d}(\mathbf{x})$ is $\Theta(nd)$;
- abcd-Formula Lower Bound: the abcd-formula complexity of $f_{n,\log n}(\mathbf{x})$ is $n^{\Theta(\log\log n)}$.

The Proof Idea:

1. Use low degree to make the abcd-formula structured.

3. Convert the structured abcd-formula into a homogeneous multilinear formula.
4. Use known lower bound against homogeneous multilinear formulas [HY11].

Proof Idea

The Statement: There is an explicit n^2-variate, degree-d abcd-polynomial $f_{n,d}(\mathbf{x})$ such that

- abcd-ABP Upper Bound: the abcd-ABP complexity of $f_{n,d}(\mathbf{x})$ is $\Theta(nd)$;
- abcd-Formula Lower Bound: the abcd-formula complexity of $f_{n,\log n}(\mathbf{x})$ is $n^{\Theta(\log \log n)}$.

The Proof Idea:

1. Use low degree to make the abcd-formula structured.
2. Use the structured formula to amplify degree while keeping the structure intact.
3. Convert the structured abcd-formula into a homogeneous multilinear formula.
4. Use known lower bound against homogeneous multilinear formulas [HY11].

Questions?

Part 2: Identity Testing and Algebraic Independence

Algebraic Independence

In the vector space $\mathbb{R}^3$ over $\mathbb{R}$,

$$(1, 0, 1) \qquad (0, 1, 0) \qquad (1, 2, 1)$$

Algebraic Independence

In the vector space $\mathbb{R}^3$ over $\mathbb{R}$,

$$1 \times (1,0,1) + 2 \times (0,1,0) - 1 \times (1,2,1) = 0$$

Algebraic Independence

In the vector space $\mathbb{R}^3$ over $\mathbb{R}$,

$$(1, 0, 1) \qquad (0, 1, 0) \qquad (1, 2, 1)$$

are linearly dependent.

Algebraic Independence

In the vector space $\mathbb{R}^3$ over $\mathbb{R}$,

$$(1, 0, 1) \qquad (0, 1, 0) \qquad (1, 2, 1)$$

are linearly dependent.

In the space of bi-variate polynomials over $\mathbb{C}$,

$$x^2 \qquad y^2 \qquad xy$$

Algebraic Independence

In the vector space $\mathbb{R}^3$ over $\mathbb{R}$,

$$(1,0,1) \qquad (0,1,0) \qquad (1,2,1)$$

are linearly dependent.

In the space of bi-variate polynomials over $\mathbb{C}$,

$$x^2 \quad \times \quad y^2 \quad - \quad (xy)^2 \quad = \quad 0$$

Algebraic Independence

In the vector space $\mathbb{R}^3$ over $\mathbb{R}$,

$$(1, 0, 1) \qquad (0, 1, 0) \qquad (1, 2, 1)$$

are linearly dependent.

In the space of bi-variate polynomials over $\mathbb{C}$,

$$x^2 \qquad y^2 \qquad xy$$

are algebraically dependent.

Algebraic Independence

A set of polynomials $\{f_1, \ldots, f_k\} \subseteq \mathbb{F}[x_1, x_2, \ldots, x_n]$ are said to be algebraically dependent if there exists $A \in \mathbb{F}[y_1, \ldots, y_k]$

Algebraic Independence

A set of polynomials $\{f_1, \ldots, f_k\} \subseteq \mathbb{F}[x_1, x_2, \ldots, x_n]$ are said to be algebraically dependent if there exists $A \in \mathbb{F}[y_1, \ldots, y_k]$ such that

$$A(y_1, \ldots, y_k) \neq 0; \qquad A(f_1, \ldots, f_k) = 0.$$

Algebraic Independence

A set of polynomials $\{f_1, \ldots, f_k\} \subseteq \mathbb{F}[x_1, x_2, \ldots, x_n]$ are said to be algebraically dependent if there exists $A \in \mathbb{F}[y_1, \ldots, y_k]$ such that

$$A(y_1, \ldots, y_k) \neq 0; \qquad A(f_1, \ldots, f_k) = 0.$$

Otherwise, they are said to be algebraically independent.

Algebraic Independence

A set of polynomials $\{f_1, \ldots, f_k\} \subseteq \mathbb{F}[x_1, x_2, \ldots, x_n]$ are said to be algebraically dependent if there exists $A \in \mathbb{F}[y_1, \ldots, y_k]$ such that

$$A(y_1, \ldots, y_k) \neq 0; \qquad A(f_1, \ldots, f_k) = 0.$$

Otherwise, they are said to be algebraically independent.

Note: The underlying field is very important.

Algebraic Independence

A set of polynomials $\{f_1, \ldots, f_k\} \subseteq \mathbb{F}[x_1, x_2, \ldots, x_n]$ are said to be algebraically dependent if there exists $A \in \mathbb{F}[y_1, \ldots, y_k]$ such that

$$A(y_1, \ldots, y_k) \neq 0; \qquad A(f_1, \ldots, f_k) = 0.$$

Otherwise, they are said to be algebraically independent.

Note: The underlying field is very important. For any prime p,

$$x^p + y^p \qquad x + y$$

Algebraic Independence

A set of polynomials $\{f_1, \ldots, f_k\} \subseteq \mathbb{F}[x_1, x_2, \ldots, x_n]$ are said to be algebraically dependent if there exists $A \in \mathbb{F}[y_1, \ldots, y_k]$ such that

$$A(y_1, \ldots, y_k) \neq 0; \qquad A(f_1, \ldots, f_k) = 0.$$

Otherwise, they are said to be algebraically independent.

Note: The underlying field is very important. For any prime p,

$$x^p + y^p \qquad x + y$$

- are algebraically independent over $\mathbb{C}$.

Algebraic Independence

A set of polynomials $\{f_1, \ldots, f_k\} \subseteq \mathbb{F}[x_1, x_2, \ldots, x_n]$ are said to be algebraically dependent if there exists $A \in \mathbb{F}[y_1, \ldots, y_k]$ such that

$$A(y_1, \ldots, y_k) \neq 0; \qquad A(f_1, \ldots, f_k) = 0.$$

Otherwise, they are said to be algebraically independent.

Note: The underlying field is very important. For any prime p,

$$x^p + y^p \qquad x + y$$

- are algebraically independent over $\mathbb{C}$.
- are algebraically dependent over $\mathbb{F}_p$,

Algebraic Independence

A set of polynomials $\{f_1, \ldots, f_k\} \subseteq \mathbb{F}[x_1, x_2, \ldots, x_n]$ are said to be algebraically dependent if there exists $A \in \mathbb{F}[y_1, \ldots, y_k]$ such that

$$A(y_1, \ldots, y_k) \neq 0; \qquad A(f_1, \ldots, f_k) = 0.$$

Otherwise, they are said to be algebraically independent.

Note: The underlying field is very important. For any prime p,

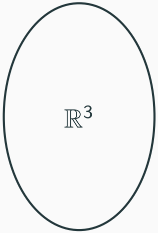
$$x^p + y^p \qquad x + y$$

- are algebraically independent over $\mathbb{C}$.
- are algebraically dependent over $\mathbb{F}_p$, since $x^p + y^p = (x+y)^p$.

Algebraic Rank

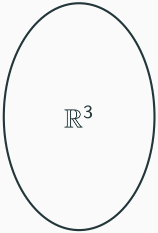
- Linear rank of $S = \{v_1, \ldots, v_m\} \subseteq \mathbb{V}$ is the size of the largest linearly independent subset of S.

Algebraic Rank



- Linear rank of $S = \{v_1, \ldots, v_m\} \subseteq \mathbb{V}$ is the size of the largest linearly independent subset of S.
- Linear rank of $\{(1,0,1), (0,1,0), (1,2,1)\}$ is 2.

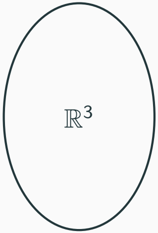
Algebraic Rank



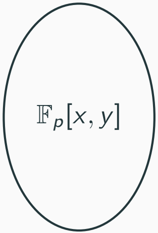
- Linear rank of $S = \{v_1, \ldots, v_m\} \subseteq \mathbb{V}$ is the size of the largest linearly independent subset of S.
- Linear rank of $\{(1,0,1),(0,1,0),(1,2,1)\}$ is 2.

- Algebraic rank of $S = \{f_1, \ldots, f_m\} \subseteq \mathbb{F}[\mathbf{x}]$ is the size of the largest algebraically independent subset of S.

Algebraic Rank

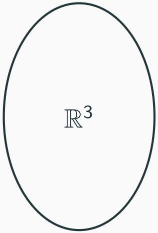


- Linear rank of $S = \{v_1, \ldots, v_m\} \subseteq \mathbb{V}$ is the size of the largest linearly independent subset of S.
- Linear rank of $\{(1,0,1),(0,1,0),(1,2,1)\}$ is 2.

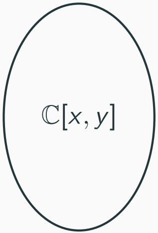


- Algebraic rank of $S = \{f_1, \ldots, f_m\} \subseteq \mathbb{F}[\mathbf{x}]$ is the size of the largest algebraically independent subset of S.
- Algebraic rank of $\{x^p + y^p, x + y\}$ is 1

Algebraic Rank



- Linear rank of $S = \{v_1, \ldots, v_m\} \subseteq \mathbb{V}$ is the size of the largest linearly independent subset of S.
- Linear rank of $\{(1,0,1),(0,1,0),(1,2,1)\}$ is 2.



- Algebraic rank of $S = \{f_1, \ldots, f_m\} \subseteq \mathbb{F}[\mathbf{x}]$ is the size of the largest algebraically independent subset of S.
- Algebraic rank of $\{x^p + y^p, x + y\}$ is 2

Rank Preserving Maps

Basis in Linear Algebra

Given a set of vectors $\{v_1, v_2, \ldots, v_m\}$ with linear rank k, there is a basis of size k.

Rank Preserving Maps

Basis in Linear Algebra

Given a set of vectors $\{v_1, v_2, \ldots, v_m\}$ with linear rank k, there is a basis of size k.

Faithful Maps

Given a set of polynomials $\{f_1, f_2, \ldots, f_m\}$ with algebraic rank k, a map

$$\varphi : \{x_1, x_2, \ldots, x_n\} \to \mathbb{F}[y_1, y_2, \ldots, y_k]$$

is said to be a faithful map if the algebraic rank of $\{f_1 \circ \varphi, f_2 \circ \varphi, \ldots, f_m \circ \varphi\}$ is also k.

Rank Preserving Maps

Basis in Linear Algebra

Given a set of vectors $\{v_1, v_2, \ldots, v_m\}$ with linear rank k, there is a basis of size k.

Faithful Maps

Given a set of polynomials $\{f_1, f_2, \ldots, f_m\}$ with algebraic rank k, a map

$$\varphi : \{x_1, x_2, \ldots, x_n\} \to \mathbb{F}[y_1, y_2, \ldots, y_k]$$

is said to be a faithful map if the algebraic rank of $\{f_1 \circ \varphi, f_2 \circ \varphi, \ldots, f_m \circ \varphi\}$ is also k.

Question: Can we construct faithful maps efficiently?

Rank Preserving Maps

Basis in Linear Algebra

Given a set of vectors $\{v_1, v_2, \ldots, v_m\}$ with linear rank k, there is a basis of size k.

Faithful Maps

Given a set of polynomials $\{f_1, f_2, \ldots, f_m\}$ with algebraic rank k, a map

$$\varphi : \{x_1, x_2, \ldots, x_n\} \to \mathbb{F}[y_1, y_2, \ldots, y_k]$$

is said to be a faithful map if the algebraic rank of $\{f_1 \circ \varphi, f_2 \circ \varphi, \ldots, f_m \circ \varphi\}$ is also k.

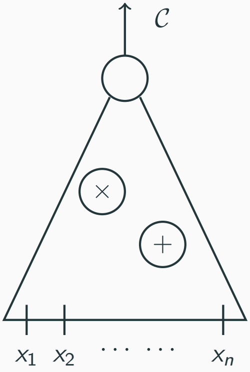
Question: Can we construct faithful maps efficiently?

Bonus: Helps in polynomial identity testing.

Polynomial Identity Testing

Given: Circuit $\mathcal{C}$ that computes an n-variate, degree d polynomial

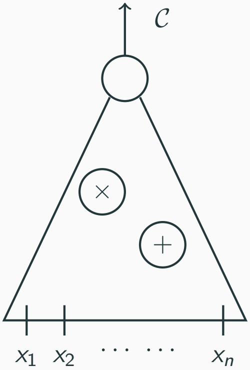
Goal: Check whether $\mathcal{C} \cong$ Zero Polynomial.



Polynomial Identity Testing

Given: Circuit $\mathcal{C}$ that computes an n-variate, degree d polynomial

Goal: Check whether $\mathcal{C} \cong$ Zero Polynomial.

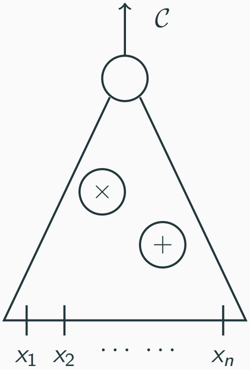


Trivial Upperbound: $(d+1)^n$

Polynomial Identity Testing

Given: Circuit $\mathcal{C}$ that computes an n-variate, degree d polynomial

Goal: Check whether $\mathcal{C} \cong$ Zero Polynomial.

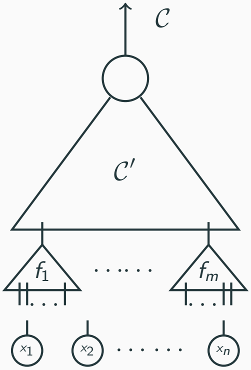


Trivial Upperbound: $(d+1)^n$

Approach: Reduce no. of variables
Keep degree under control
Preserve non-zeroness

Polynomial Identity Testing

Given: Circuit $\mathcal{C}$ that computes an n-variate, degree d polynomial
Goal: Check whether $\mathcal{C} \cong$ Zero Polynomial.



Trivial Upperbound: $(d+1)^n$

Approach: Reduce no. of variables
Keep degree under control
Preserve non-zeroness

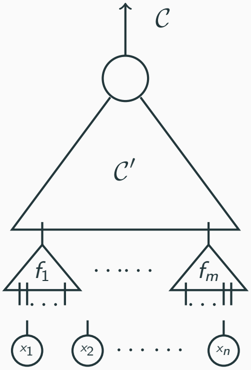
Special Case: $\mathcal{C} = \mathcal{C}'(f_1, f_2, \ldots, f_m)$ where algebraic rank of $\{f_1, \ldots, f_m\} = k$, and

$$k \ll n$$

Polynomial Identity Testing

Given: Circuit $\mathcal{C}$ that computes an n-variate, degree d polynomial

Goal: Check whether $\mathcal{C} \cong$ Zero Polynomial.



Trivial Upperbound: $(d+1)^n$

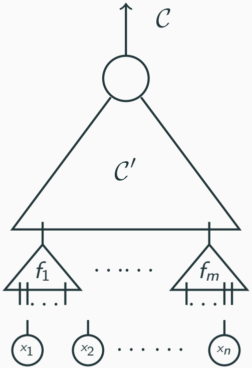
Approach: Reduce no. of variables
Keep degree under control
Preserve non-zeroness

Special Case: $\mathcal{C} = \mathcal{C}'(f_1, f_2, \ldots, f_m)$ where algebraic rank of $\{f_1, \ldots, f_m\} = k$, and

$$k \ll n$$

Question: Can the upperbound be made $\approx (d+1)^k$?

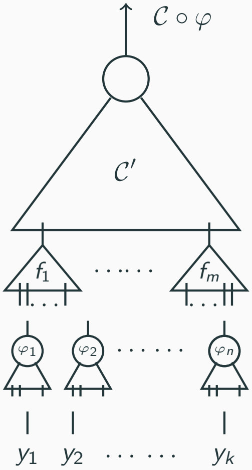
Faithful Maps & Polynomial Identity Testing



Question: Suppose $\mathcal{C} = \mathcal{C}'(f_1, f_2, \ldots, f_m)$ where algebraic rank of $\{f_1, \ldots, f_m\} = k$, and $k \ll n$.

Can the upper bound be made $\approx (d+1)^k$?

Faithful Maps & Polynomial Identity Testing

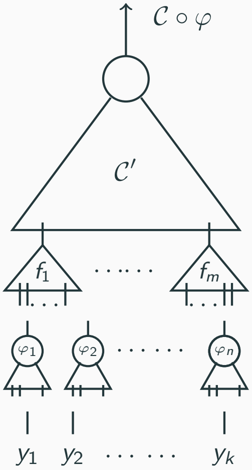


Question: Suppose $\mathcal{C} = \mathcal{C}'(f_1, f_2, \ldots, f_m)$ where algebraic rank of $\{f_1, \ldots, f_m\} = k$, and $k \ll n$.
Can the upper bound be made $\approx (d+1)^k$?

[Beecken-Mittman-Saxena, Agrawal-Saha-Saptharishi-Saxena]

If $\varphi : \{x_!, \ldots, x_n\} \to \mathbb{F}[y_1, \ldots, y_k]$ is a faithful map, then

Faithful Maps & Polynomial Identity Testing



Question: Suppose $\mathcal{C} = \mathcal{C}'(f_1, f_2, \dots, f_m)$ where algebraic rank of $\{f_1, \dots, f_m\} = k$, and $k \ll n$.
Can the upper bound be made $\approx (d+1)^k$?

[Beecken-Mittman-Saxena, Agrawal-Saha-Saptharishi-Saxena]

If $\varphi : \{x_!, \dots, x_n\} \to \mathbb{F}[y_1, \dots, y_k]$ is a faithful map, then

$$\mathcal{C}(f_1, f_2, \dots, f_m) \neq 0 \text{ if and only if}$$

$$(\mathcal{C}(f_1(\varphi), f_2(\varphi), \dots f_m(\varphi))) \neq 0.$$

The Question

Given a set of polynomials $\{f_1, f_2, \ldots, f_m\} \subseteq \mathbb{F}[x_1, \ldots, x_n]$, we want to construct a map

$$\varphi : \{x_1, x_2, \ldots, x_n\} \to \mathbb{F}[y_1, y_2, \ldots, y_k]$$

such that

$$\text{algrank}(f_1(\varphi), f_2(\varphi), \ldots, f_m(\varphi)) = \text{algrank}(f_1, f_2, \ldots, f_m)$$

The Question

Given a set of polynomials $\{f_1, f_2, \ldots, f_m\} \subseteq \mathbb{F}[x_1, \ldots, x_n]$, we want to construct a map

$$\varphi : \{x_1, x_2, \ldots, x_n\} \to \mathbb{F}[y_1, y_2, \ldots, y_k]$$

such that

$$\text{algrank}(f_1(\varphi), f_2(\varphi), \ldots, f_m(\varphi)) = \text{algrank}(f_1, f_2, \ldots, f_m)$$

Fact: A random affine transformation is a faithful map

$$\varphi : x_i = \sum_{j=1}^{k} s_{ij} y_j + a_i$$

The Question

Given a set of polynomials $\{f_1, f_2, \dots, f_m\} \subseteq \mathbb{F}[x_1, \dots, x_n]$, we want to construct a map

$$\varphi : \{x_1, x_2, \dots, x_n\} \to \mathbb{F}[y_1, y_2, \dots, y_k]$$

such that

$$\text{algrank}(f_1(\varphi), f_2(\varphi), \dots, f_m(\varphi)) = \text{algrank}(f_1, f_2, \dots, f_m)$$

Fact: A random affine transformation is a faithful map

$$\varphi : x_i = \sum_{j=1}^{k} s_{ij} y_j + a_i$$

Question: Can we construct faithful maps deterministically?

Characteristic Zero Fields [B-M-S, A-S-S-S]

Step 1: Capture algebraic rank via linear rank

Characteristic Zero Fields [B-M-S, A-S-S-S]

Step 1: Capture algebraic rank via linear rank

For $\{f_1, f_2, \ldots, f_m\} \subseteq \mathbb{F}[x_1, x_2, \ldots, x_n]$ and $\mathbf{f} = (f_1, f_2, \ldots, f_m)$,

$$\mathbf{J_x}(\mathbf{f}) = \begin{bmatrix} \partial_{x_1}(f_1) & \partial_{x_1}(f_2) & \ldots & \partial_{x_1}(f_n) \\ \partial_{x_2}(f_1) & \partial_{x_2}(f_2) & \ldots & \partial_{x_2}(f_n) \\ \vdots & \vdots & \ddots & \vdots \\ \partial_{x_n}(f_1) & \partial_{x_n}(f_2) & \ldots & \partial_{x_n}(f_m) \end{bmatrix}$$

Characteristic Zero Fields [B-M-S, A-S-S-S]

Step 1: Capture algebraic rank via linear rank

For $\{f_1, f_2, \ldots, f_m\} \subseteq \mathbb{F}[x_1, x_2, \ldots, x_n]$ and $\mathbf{f} = (f_1, f_2, \ldots, f_m)$,

$$\mathbf{J_x}(\mathbf{f}) = \begin{bmatrix} \partial_{x_1}(f_1) & \partial_{x_1}(f_2) & \ldots & \partial_{x_1}(f_n) \\ \partial_{x_2}(f_1) & \partial_{x_2}(f_2) & \ldots & \partial_{x_2}(f_n) \\ \vdots & \vdots & \ddots & \vdots \\ \partial_{x_n}(f_1) & \partial_{x_n}(f_2) & \ldots & \partial_{x_n}(f_m) \end{bmatrix}$$

The Jacobian Criterion [Jacobi]

If $\mathbb{F}$ has characteristic zero, the algebraic rank of $\{f_1, f_2, \ldots, f_m\}$ is equal to the linear rank of its Jacobian matrix.

Characteristic Zero Fields [B-M-S, A-S-S-S]

Step 1: Capture algebraic rank via linear rank of the Jacobian

For $\{f_1, f_2, \ldots, f_m\} \subseteq \mathbb{F}[x_1, x_2, \ldots, x_n]$ and $\mathbf{f} = (f_1, f_2, \ldots, f_m)$,

$$\mathbf{J_x}(\mathbf{f}) = \begin{bmatrix} \partial_{x_1}(f_1) & \partial_{x_1}(f_2) & \ldots & \partial_{x_1}(f_n) \\ \partial_{x_2}(f_1) & \partial_{x_2}(f_2) & \ldots & \partial_{x_2}(f_n) \\ \vdots & \vdots & \ddots & \vdots \\ \partial_{x_n}(f_1) & \partial_{x_n}(f_2) & \ldots & \partial_{x_n}(f_m) \end{bmatrix}$$

The Jacobian Criterion [Jacobi]

If $\mathbb{F}$ has characteristic zero, the algebraic rank of $\{f_1, f_2, \ldots, f_m\}$ is equal to the linear rank of its Jacobian matrix.

Characteristic Zero Fields [B-M-S, A-S-S-S]

Step 2: Start with a generic linear transformation. $\varphi : x_i = \sum_{j=1}^{k} s_{ij} y_j + a_i$

Characteristic Zero Fields [B-M-S, A-S-S-S]

Step 2: Start with a generic linear transformation. $\varphi : x_i = \sum_{j=1}^{k} s_{ij} y_j + a_i$

$$\begin{bmatrix} \\ \mathbf{J_y}(\mathbf{f}(\varphi)) \\ \\ \end{bmatrix}$$

Characteristic Zero Fields [B-M-S, A-S-S-S]

Step 2: Start with a generic linear transformation. $\varphi : x_i = \sum_{j=1}^{k} s_{ij} y_j + a_i$

$$\left[\ \mathbf{J_y}(\mathbf{f}(\varphi))\ \right] = \left[\ \varphi(\mathbf{J_x}(\mathbf{f}))\ \right] \times \left[\ M_\varphi\ \right]$$

Characteristic Zero Fields [B-M-S, A-S-S-S]

Step 2: Start with a generic linear transformation. $\varphi : x_i = \sum_{j=1}^{k} s_{ij} y_j + a_i$

$$\left[\quad \mathbf{J_y}(\mathbf{f}(\varphi)) \quad \right] = \left[\quad \varphi(\mathbf{J_x}(\mathbf{f})) \quad \right] \times \left[\quad M_\varphi \quad \right]$$

What we need: φ such that

- $\mathrm{rank}(\mathbf{J_x}(\mathbf{f})) = \mathrm{rank}(\varphi(\mathbf{J_x}(\mathbf{f})))$

Characteristic Zero Fields [B-M-S, A-S-S-S]

Step 2: Start with a generic linear transformation. $\varphi : x_i = \sum_{j=1}^{k} s_{ij} y_j + a_i$

$$\begin{bmatrix} \\ \mathbf{J_y}(\mathbf{f}(\varphi)) \\ \\ \end{bmatrix} = \begin{bmatrix} \\ \varphi(\mathbf{J_x}(\mathbf{f})) \\ \\ \end{bmatrix} \times \begin{bmatrix} \\ \\ M_\varphi \\ \\ \\ \end{bmatrix}$$

What we need: φ such that

- $\text{rank}(\mathbf{J_x}(\mathbf{f})) = \text{rank}(\varphi(\mathbf{J_x}(\mathbf{f})))$: Can be done if f_is are structured (a_is are responsible)

Characteristic Zero Fields [B-M-S, A-S-S-S]

Step 2: Start with a generic linear transformation. $\varphi : x_i = \sum_{j=1}^{k} s_{ij} y_j + a_i$

$$\left[\mathbf{J_y}(\mathbf{f}(\varphi)) \right] = \left[\varphi(\mathbf{J_x}(\mathbf{f})) \right] \times \left[M_\varphi \right]$$

What we need: φ such that

- $\text{rank}(\mathbf{J_x}(\mathbf{f})) = \text{rank}(\varphi(\mathbf{J_x}(\mathbf{f})))$: Can be done if f_is are structured (a_is are responsible)
- M_φ preserves rank

A Rank Preserving Matrix and a Faithful Map [BMS13]

$$\varphi : x_i = \sum_{j=1}^{k} s_{ij} y_j + a_i$$

Chain Rule $\Rightarrow M_\varphi[i,j] = s_{ij}$

A Rank Preserving Matrix and a Faithful Map [BMS13]

$$\varphi : x_i = \sum_{j=1}^{k} s_{ij} y_j + a_i$$

$$\text{Chain Rule} \Rightarrow M_\varphi[i,j] = s_{ij}$$

For every $m \times n$ matrix A, $\operatorname{rank}(A) = \operatorname{rank}(AM_\varphi)$.

A Rank Preserving Matrix and a Faithful Map [BMS13]

$$\varphi : x_i = \sum_{j=1}^{k} s_{ij} y_j + a_i$$

$$\text{Chain Rule} \Rightarrow M_\varphi[i, j] = s_{ij}$$

For every $m \times n$ matrix A, $\text{rank}(A) = \text{rank}(AM_\varphi)$.

[GR05]: Vandermonde type matrices preserve rank.

$$\begin{bmatrix} s & s^2 & \dots & s^k \\ s^2 & s^4 & \dots & s^{2k} \\ \vdots & \vdots & & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & & \vdots \\ s^n & s^{2n} & \dots & s^{kn} \end{bmatrix}$$

A Rank Preserving Matrix and a Faithful Map [BMS13]

$$\varphi : x_i = \sum_{j=1}^{k} s_{ij} y_j + a_i$$

$$\text{Chain Rule} \Rightarrow M_\varphi[i,j] = s_{ij}$$

For every $m \times n$ matrix A, $\text{rank}(A) = \text{rank}(AM_\varphi)$.

Family of matrices or one matrix parameterised by s: $\left\{M_{\varphi(s)}\right\}_{s \in \mathcal{F}}$

[GR05]: Vandermonde type matrices preserve rank.

$$\begin{bmatrix} s & s^2 & \dots & s^k \\ s^2 & s^4 & \dots & s^{2k} \\ \vdots & \vdots & & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & & \vdots \\ s^n & s^{2n} & \dots & s^{kn} \end{bmatrix}$$

A Rank Preserving Matrix and a Faithful Map [BMS13]

$$\varphi : x_i = \sum_{j=1}^{k} s_{ij} y_j + a_i$$

Chain Rule $\Rightarrow M_\varphi[i,j] = s_{ij}$

For every $m \times n$ matrix A, $\text{rank}(A) = \text{rank}(AM_\varphi)$.

Family of matrices or one matrix parameterised by s: $\left\{M_{\varphi(s)}\right\}_{s \in \mathcal{F}}$

$$\varphi : x_i = \sum_{j=1}^{k} s^{ij} y_j + a_i \text{ will work.}$$

[GR05]: Vandermonde type matrices preserve rank.

$$\begin{bmatrix} s & s^2 & \dots & s^k \\ s^2 & s^4 & \dots & s^{2k} \\ \vdots & \vdots & & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & & \vdots \\ s^n & s^{2n} & \dots & s^{kn} \end{bmatrix}$$

What happens over Finite Characteristic Fields?

The Jacobian Criterion is **false** over finite characteristic fields.

What happens over Finite Characteristic Fields?

The Jacobian Criterion is **false** over finite characteristic fields.

Taylor Expansion: For any $f \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ and $\mathbf{z} \in \mathbb{F}^n$,

$$f(\mathbf{x} + \mathbf{z}) - f(\mathbf{z}) = \underbrace{x_1 \cdot \partial_{x_1} f + \cdots + x_n \cdot \partial_{x_n} f}_{\text{Jacobian}} + \text{higher order terms}$$

What happens over Finite Characteristic Fields?

The Jacobian Criterion is **false** over finite characteristic fields.

Taylor Expansion: For any $f \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ and $\mathbf{z} \in \mathbb{F}^n$,

$$f(\mathbf{x} + \mathbf{z}) - f(\mathbf{z}) = \underbrace{x_1 \cdot \partial_{x_1} f + \cdots + x_n \cdot \partial_{x_n} f}_{\text{Jacobian}} + \text{higher order terms}$$

[Pandey-Saxena-Sinhababu]: Look up till the inseparable degree in the expansion.

What happens over Finite Characteristic Fields?

The Jacobian Criterion is **false** over finite characteristic fields.

Taylor Expansion: For any $f \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ and $\mathbf{z} \in \mathbb{F}^n$,

$$f(\mathbf{x} + \mathbf{z}) - f(\mathbf{z}) = \underbrace{x_1 \cdot \partial_{x_1} f + \cdots + x_n \cdot \partial_{x_n} f}_{\text{Jacobian}} + \text{higher order terms}$$

[Pandey-Saxena-Sinhababu]: Look up till the inseparable degree in the expansion.

A New Operator: For any $f \in \mathbb{F}[x_1, x_2, \ldots, x_n]$,

$$\mathcal{H}_t(f) = \deg^{\leq t}(f(\mathbf{x} + \mathbf{z}) - f(\mathbf{z}))$$

What happens over Finite Characteristic Fields?

The Jacobian Criterion is **false** over finite characteristic fields.

Taylor Expansion: For any $f \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ and $\mathbf{z} \in \mathbb{F}^n$,

$$f(\mathbf{x}+\mathbf{z}) - f(\mathbf{z}) = \underbrace{x_1 \cdot \partial_{x_1} f + \cdots + x_n \cdot \partial_{x_n} f}_{\text{Jacobian}} + \text{higher order terms}$$

[Pandey-Saxena-Sinhababu]: Look up till the inseparable degree in the expansion.

A New Operator: For any $f \in \mathbb{F}[x_1, x_2, \ldots, x_n]$,

$$\mathcal{H}_t(f) = \deg^{\leq t}\left(f(\mathbf{x}+\mathbf{z}) - f(\mathbf{z})\right)$$

$$\hat{\mathcal{H}}(\mathbf{f}) = \begin{bmatrix} \ldots & \mathcal{H}_t(f_1) & \ldots \\ \ldots & \mathcal{H}_t(f_2) & \ldots \\ & \vdots & \\ \ldots & \mathcal{H}_t(f_k) & \ldots \end{bmatrix}$$

Alternate Criterion for the General Case [Pandey-Saxena-Sinhababu]

$f_1, f_2, \ldots, f_k \in \mathbb{F}[\mathbf{x}]$ are algebraically independent if and only if for every $(v_1, v_2, \ldots, v_k)$ with v_is in $\mathcal{I}_t$,

Alternate Criterion for the General Case [Pandey-Saxena-Sinhababu]

$f_1, f_2, \ldots, f_k \in \mathbb{F}[\mathbf{x}]$ are algebraically independent if and only if for every $(v_1, v_2, \ldots, v_k)$ with v_is in $\mathcal{I}_t$,

$$\mathcal{H}(\mathbf{f}, \mathbf{v}) = \begin{bmatrix} \ldots & \mathcal{H}_t(f_1) + v_1 & \ldots \\ \ldots & \mathcal{H}_t(f_2) + v_2 & \ldots \\ & \vdots & \\ \ldots & \mathcal{H}_t(f_k) + v_k & \ldots \end{bmatrix}$$

Alternate Criterion for the General Case [Pandey-Saxena-Sinhababu]

$f_1, f_2, \ldots, f_k \in \mathbb{F}[\mathbf{x}]$ are algebraically independent if and only if for every $(v_1, v_2, \ldots, v_k)$ with v_is in $\mathcal{I}_t$,

$$\mathcal{H}(\mathbf{f}, \mathbf{v}) = \begin{bmatrix} \ldots & \mathcal{H}_t(f_1) + v_1 & \ldots \\ \ldots & \mathcal{H}_t(f_2) + v_2 & \ldots \\ & \vdots & \\ \ldots & \mathcal{H}_t(f_k) + v_k & \ldots \end{bmatrix} \text{ has full rank over } \mathbb{F}(\mathbf{z})$$

Alternate Criterion for the General Case [Pandey-Saxena-Sinhababu]

$f_1, f_2, \ldots, f_k \in \mathbb{F}[\mathbf{x}]$ are algebraically independent if and only if for every $(v_1, v_2, \ldots, v_k)$ with v_is in $\mathcal{I}_t$,

$$\mathcal{H}(\mathbf{f}, \mathbf{v}) = \begin{bmatrix} \ldots & \mathcal{H}_t(f_1) + v_1 & \ldots \\ \ldots & \mathcal{H}_t(f_2) + v_2 & \ldots \\ & \vdots & \\ \ldots & \mathcal{H}_t(f_k) + v_k & \ldots \end{bmatrix} \text{ has full rank over } \mathbb{F}(\mathbf{z})$$

where t is the inseparable degree of $\{f_1, f_2, \ldots, f_k\}$ and

$$\mathcal{I}_t = \langle \mathcal{H}_t(f_1), \mathcal{H}_t(f_2), \ldots, \mathcal{H}_t(f_k) \rangle_{\mathbb{F}(\mathbf{z})}^{\geq 2} \bmod \langle \mathbf{x} \rangle^{t+1} \subseteq \mathbb{F}(\mathbf{z})[\mathbf{x}].$$

Our Result

Suppose
- $\circ$ $f_1, \ldots, f_m \in \mathbb{F}[x_1, \ldots, x_n]$
- $\circ$ algebraic rank of $\{f_1, \ldots, f_m\} = k$
- $\circ$ inseparable degree of $\{f_1, \ldots, f_m\} = t$

Our Result

Suppose
- $f_1, \ldots, f_m \in \mathbb{F}[x_1, \ldots, x_n]$
- algebraic rank of $\{f_1, \ldots, f_m\} = k$
- inseparable degree of $\{f_1, \ldots, f_m\} = t$

Then, we can construct

$$\Phi : \mathbb{F}[\mathbf{x}] \to \mathbb{F}(s)[y_0, y_1, \ldots, y_k]$$

Our Result

Suppose

- $\circ$ $f_1, \ldots, f_m \in \mathbb{F}[x_1, \ldots, x_n]$
- $\circ$ algebraic rank of $\{f_1, \ldots, f_m\} = k$
- $\circ$ inseparable degree of $\{f_1, \ldots, f_m\} = t$

Then, we can construct

$$\Phi : \mathbb{F}[\mathbf{x}] \to \mathbb{F}(s)[y_0, y_1, \ldots, y_k]$$

such that

$$\mathrm{algrank}_{\mathbb{F}}(f_1 \circ \Phi, \ldots, f_m \circ \Phi) = k$$

Our Result

Suppose
- ∘ $f_1, \ldots, f_m \in \mathbb{F}[x_1, \ldots, x_n]$
- ∘ algebraic rank of $\{f_1, \ldots, f_m\} = k$
- ∘ inseparable degree of $\{f_1, \ldots, f_m\} = t$

Then, we can construct

$$\Phi : \mathbb{F}[\mathbf{x}] \to \mathbb{F}(s)[y_0, y_1, \ldots, y_k]$$

such that

$$\text{algrank}_{\mathbb{F}}(f_1 \circ \Phi, \ldots, f_m \circ \Phi) = k$$

whenever

- each of the f_i's are sparse polynomials,
- each of the f_i's are products of variable disjoint, multilinear, sparse polynomials.

Proof Overview

Step 1: Capture algebraic rank via linear rank of the PSS-Jacobian

Proof Overview

Step 1: Capture algebraic rank via linear rank of the PSS-Jacobian

Step 2: For a *generic linear map* $\Phi : \mathbf{x} \to \mathbb{F}(s)[y_1, \ldots, y_k]$, write **PSS** $\mathbf{J_y}(\mathbf{f} \circ \Phi)$ in terms of **PSS** $\mathbf{J_x}(\mathbf{f})$.

Proof Overview

Step 1: Capture algebraic rank via linear rank of the PSS-Jacobian

Step 2: For a *generic linear map* $\Phi : \mathbf{x} \to \mathbb{F}(s)[y_1, \ldots, y_k]$, write $\mathbf{PSS\ J_y(f \circ \Phi)}$ in terms of $\mathbf{PSS\ J_x(f)}$. This can be described succinctly as

$$\mathbf{PSS\ J_y}(f \circ \Phi) = \Phi(\mathbf{PSS\ J_x(f)}) \cdot M_\Phi.$$

Proof Overview

Step 1: Capture algebraic rank via linear rank of the PSS-Jacobian

Step 2: For a *generic linear map* $\Phi : \mathbf{x} \to \mathbb{F}(s)[y_1, \ldots, y_k]$, write $\textbf{PSS } \mathbf{J}_{\mathbf{y}}(\mathbf{f} \circ \Phi)$ in terms of $\textbf{PSS } \mathbf{J}_{\mathbf{x}}(\mathbf{f})$. This can be described succinctly as

$$\textbf{PSS } \mathbf{J}_{\mathbf{y}}(f \circ \Phi) = \Phi(\textbf{PSS } \mathbf{J}_{\mathbf{x}}(\mathbf{f})) \cdot M_{\Phi}.$$

What we need: Φ such that

- $\mathrm{rank}(\Phi(\textbf{PSS } \mathbf{J}_{\mathbf{x}}(\mathbf{f}))) = \mathrm{rank}(\textbf{PSS } \mathbf{J}_{\mathbf{x}}(\mathbf{f}))$: Can be done if $\mathbf{f}$'s are some structured polynomials (for example, sparse).

Proof Overview

Step 1: Capture algebraic rank via linear rank of the PSS-Jacobian

Step 2: For a *generic linear map* $\Phi : \mathbf{x} \to \mathbb{F}(s)[y_1, \ldots, y_k]$, write $\textbf{PSS } \mathbf{J_y}(\mathbf{f} \circ \Phi)$ in terms of $\textbf{PSS } \mathbf{J_x}(\mathbf{f})$. This can be described succinctly as

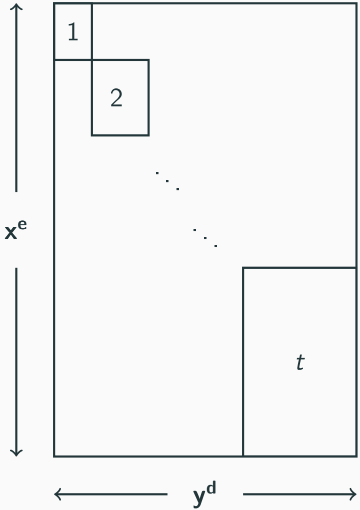
$$\textbf{PSS } \mathbf{J_y}(f \circ \Phi) = \Phi(\textbf{PSS } \mathbf{J_x}(\mathbf{f})) \cdot M_\Phi.$$

What we need: Φ such that

- $\mathrm{rank}(\Phi(\textbf{PSS } \mathbf{J_x}(\mathbf{f}))) = \mathrm{rank}(\textbf{PSS } \mathbf{J_x}(\mathbf{f}))$: Can be done if $\mathbf{f}$'s are some structured polynomials (for example, sparse).
- M_Φ preserves rank. That is,

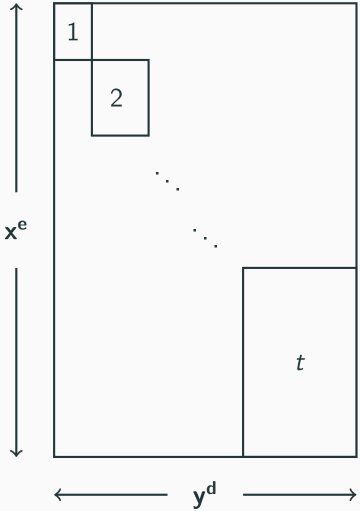
$$\mathrm{rank}(\Phi(\textbf{PSS } \mathbf{J_x}(\mathbf{f})) \cdot M_\Phi) = \mathrm{rank}(\Phi(\textbf{PSS } \mathbf{J_x}(\mathbf{f}))).$$

The Faithful Map



$$M_\Phi(\mathbf{x}^{\mathbf{e}}, \mathbf{y}^{\mathbf{d}}) = \mathrm{coeff}_{\mathbf{y}^{\mathbf{d}}}(\Phi(\mathbf{x}^{\mathbf{e}}))$$

The Faithful Map

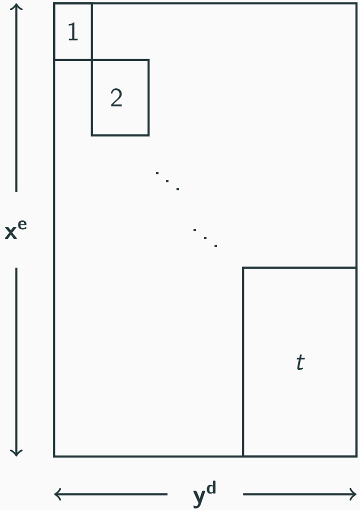


$$M_\Phi(\mathbf{x}^{\mathbf{e}}, \mathbf{y}^{\mathbf{d}}) = \mathrm{coeff}_{\mathbf{y}^{\mathbf{d}}}(\Phi(\mathbf{x}^{\mathbf{e}}))$$

Taking inspiration from the previous case:

$$M_\Phi(x_i, y_j) = s^{\mathrm{wt}(i)j}$$

The Faithful Map



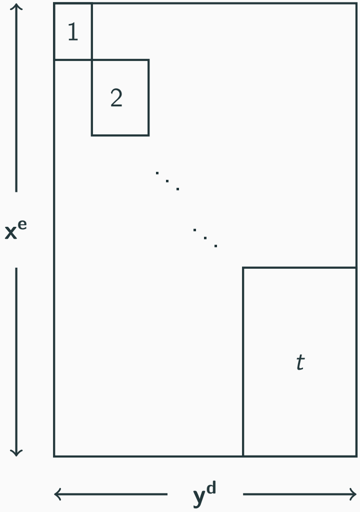
$$M_\Phi(\mathbf{x^e}, \mathbf{y^d}) = \mathrm{coeff}_{\mathbf{y^d}}(\Phi(\mathbf{x^e}))$$

Taking inspiration from the previous case:

$$M_\Phi(x_i, y_j) = s^{\mathrm{wt}(i)j}$$

For the correct definition of $\mathrm{wt}(i)$, things work out.

The Faithful Map



$$M_\Phi(\mathbf{x^e}, \mathbf{y^d}) = \mathrm{coeff}_{\mathbf{y^d}}(\Phi(\mathbf{x^e}))$$

Taking inspiration from the previous case:

$$M_\Phi(x_i, y_j) = s^{\mathrm{wt}(i)j}$$

For the correct definition of $\mathrm{wt}(i)$, things work out.

$$\Phi(x_i) = a_i \cdot y_0 + \sum_{j \in [k]} s^{\mathrm{wt}(i)j} \cdot y_j$$

Questions?

Thankyou!