Hitting-sets & Lower Bounds using algebraic independence

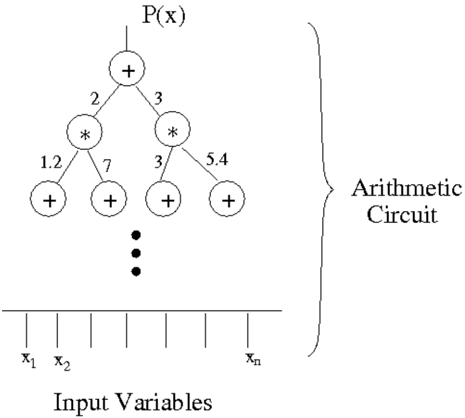
M Agrawal C Saha R Saptharishi N Saxena

Mysore Park Workshop 2012

Outline

Outline

Arithmetic Circuit



An arithmetic circuit 'computes' a polynomial $P(\mathbf{x})$ in the variables $x_1, \ldots, x_n$.

Arithmetic circuit complexity

"Study of arithmetic circuits"

The two extremes...

- Efficient algorithms: Which algorithmic questions on arithmetic circuits can be resolved efficiently?
- Lower bounds: Which polynomials do not admit small circuit representations? (formally, known as the "VP vs. VNP" question)

Arithmetic circuit complexity

"Study of arithmetic circuits"

The two extremes...

- Efficient algorithms: Which algorithmic questions on arithmetic circuits can be resolved efficiently?
- Lower bounds: Which polynomials do not admit small circuit representations? (formally, known as the "VP vs. VNP" question)

Identity Testing

PIT: A problem of prime importance in arithmetic complexity

Given an arithmetic circuit $\mathcal{C}$, test if the output $P(\mathbf{x}) \equiv 0$.

Complexity of PIT:

- Size of a circuit: s = number of gates & wires in $\mathcal{C}$.
- An identity test runs in polynomial time if its time complexity is $s^{O(1)}$.

Identity Testing

PIT: A problem of prime importance in arithmetic complexity

Given an arithmetic circuit $\mathcal{C}$, test if the output $P(\mathbf{x}) \equiv 0$.

Complexity of PIT:

- Size of a circuit: $s =$ number of gates & wires in $\mathcal{C}$.
- An identity test runs in polynomial time if its time complexity is $s^{O(1)}$.

Motivations

Why is identity testing interesting?

- Has applications in primality testing, bipartite matching, polynomial interpolation, solvability, learning etc.
- Appears in the proofs of important complexity theory results like IP $=$ PSPACE, and the PCP theorem.
- 'Derandomizing PIT' $\Rightarrow$ VP $\neq$ VNP.

Skyum & Valiant (1985):

VP $\stackrel{?}{=}$ VNP must necessarily be resolved before resolving P $\stackrel{?}{=}$ NP

A simple randomized PIT algorithm

- Identity testing can be solved in randomized polynomial time.
  - Pick a random point from $\mathbb{F}^n$ and substitute in place of $x_1, \ldots, x_n$. (Schwartz-Zippel test)

Roots are far fewer than non-roots.

Hitting sets: a 'blackbox' derandomization

Definition

A polynomial-time hitting set generator for a circuit family outputs a 'small' collection of points such that every non-zero circuit in the family evaluates to non-zero at one of the points in the collection.

'Derandomize' PIT $\overset{\text{means}}{\longrightarrow}$ design a poly-time hitting set generator.

poly-time = polynomial in the size of circuits (in the family)

Hitting sets: a 'blackbox' derandomization

Definition

A polynomial-time hitting set generator for a circuit family outputs a 'small' collection of points such that every non-zero circuit in the family evaluates to non-zero at one of the points in the collection.

'Derandomize' PIT $\stackrel{\text{means}}{\longrightarrow}$ design a poly-time hitting set generator.

poly-time = polynomial in the size of circuits (in the family)

The dual worlds: Hitting sets & lower bounds

Heintz & Schnorr (1980), Kabanets et al.(2003), Agrawal(2005), Agrawal & Vinay(2008):

Designing a poly-time hitting set generator $\overset{nearly}{\Leftrightarrow}$ proving circuit lower bounds (VP $\neq$ VNP).

"Identity testing and lower bounds are 'equally hard' problems."

The dual worlds: Hitting sets & lower bounds

Heintz & Schnorr (1980), Kabanets et al.(2003), Agrawal(2005), Agrawal & Vinay(2008):

Designing a poly-time hitting set generator $\overset{\textit{nearly}}{\Leftrightarrow}$ proving circuit lower bounds (VP $\neq$ VNP).

"Identity testing and lower bounds are 'equally hard' problems."

Depth 4 circuits: the final frontier

Agrawal (2005), Agrawal & Vinay (2008), Kabanets & Impagliazzo (2004):

- A poly-time **hitting set** generator for depth-4 circuits

  ⇒ an exponential lower bound for depth-4 circuits [1]

  ⇒ an exponential **lower bound** for general circuits

  ⇒ a **quasi-poly** time **hitting set** generator for general circuits.

[1] with some degree retrictions on multiplication gates

Depth 4 circuits: the final frontier

Agrawal (2005), Agrawal & Vinay (2008), Kabanets & Impagliazzo (2004):

- A poly-time **hitting set** generator for depth-4 circuits

  $\Rightarrow$ an exponential lower bound for depth-4 circuits [1]

  $\Rightarrow$ an exponential **lower bound** for general circuits

  $\Rightarrow$ a **quasi-poly** time **hitting set** generator for general circuits.

[1] with some degree retrictions on multiplication gates

Depth 4 circuits: the final frontier

Agrawal (2005), Agrawal & Vinay (2008), Kabanets & Impagliazzo (2004):

- A poly-time **hitting set** generator for depth-4 circuits

  $\Rightarrow$ an exponential lower bound for depth-4 circuits [1]

  $\Rightarrow$ an exponential **lower bound** for general circuits

  $\Rightarrow$ a **quasi-poly** time **hitting set** generator for general circuits.

[1] with some degree retrictions on multiplication gates

Depth 4 circuits: the final frontier

Agrawal (2005), Agrawal & Vinay (2008), Kabanets & Impagliazzo (2004):

- A poly-time **hitting set** generator for depth-4 circuits

  $\Rightarrow$ an exponential lower bound for depth-4 circuits [1]

  $\Rightarrow$ an exponential **lower bound** for general circuits

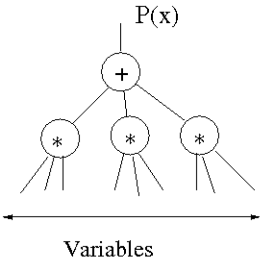
  $\Rightarrow$ a **quasi-poly** time **hitting set** generator for general circuits.

[1] with some degree retrictions on multiplication gates

Restricted models

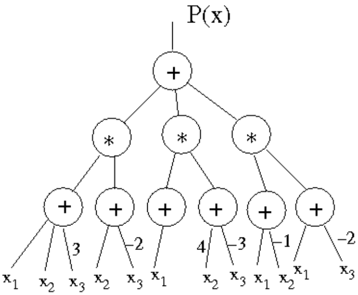
If the depth-4 case is hard to solve, why not start with depth-2, or depth-3, or restricted versions of depth-4 circuits?

Depth-2 circuits



$P(\mathbf{x})$ = sum of monomials (sparse polynomial)

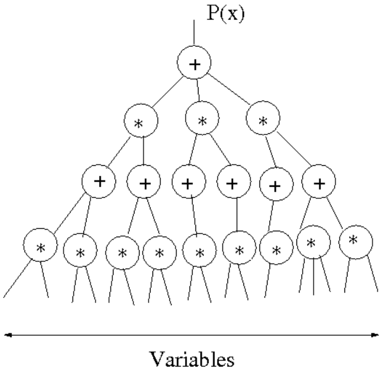
Depth-3 circuits



$P(\mathbf{x}) = \sum_{i=1}^{m} \prod_{j=1}^{d} \ell_{ij}$ (ℓ_{ij}'s are linear forms)

$m \rightarrow$ top fan-in

Depth-4 circuits



$P(\mathbf{x}) = \sum_{i=1}^{m} \prod_{j=1}^{d} P_{ij}$ ($P'_{ij}s$ are sparse polynomials)

$m \longrightarrow$ top fan-in

Known results

Three main results:

- Klivans & Spielman (2001): Poly-time hitting set generator for depth-2 circuits. (depth-2 PIT is completely resolved!)
- Saxena & Seshadhri (2011): Poly-time hitting set generator for depth-3 **constant top fan-in** circuits.
- Anderson et al. (2011): Poly-time hitting set generator for depth-4 (constant depth) **constant read** multilinear formula.

Known results

Three main results:

- Klivans & Spielman (2001): Poly-time hitting set generator for depth-2 circuits. (depth-2 PIT is completely resolved!)
- Saxena & Seshadhri (2011): Poly-time hitting set generator for depth-3 **constant top fan-in** circuits.
- Anderson et al. (2011): Poly-time hitting set generator for depth-4 (constant depth) **constant read** multilinear formula.

Known results

Three main results:

- Klivans & Spielman (2001): Poly-time hitting set generator for depth-2 circuits. (depth-2 PIT is completely resolved!)
- Saxena & Seshadhri (2011): Poly-time hitting set generator for depth-3 **constant top fan-in** circuits.
- Anderson et al. (2011): Poly-time hitting set generator for depth-4 (constant depth) **constant read** multilinear formula.

Some terminologies

multilinear polynomial → max. degree of every variable in every monomial in 1.

multilinear circuits → every gate computes a multilinear polynomial.

constant read formula → every variable occurs constantly many times at the leaves of the formula.

Some terminologies

multilinear polynomial $\longrightarrow$ max. degree of every variable in every monomial in 1.

multilinear circuits $\longrightarrow$ every gate computes a multilinear polynomial.

constant read formula $\longrightarrow$ every variable occurs constantly many times at the leaves of the formula.

Some terminologies

multilinear polynomial $\longrightarrow$ max. degree of every variable in every monomial in 1.

multilinear circuits $\longrightarrow$ every gate computes a multilinear polynomial.

constant read formula $\longrightarrow$ every variable occurs constantly many times at the leaves of the formula.

Known results: Constant top fanin depth-3 circuits

Earlier work:
Dvir & Sphilka (STOC 2005), Kayal & Saxena (CCC 2006), Saxena & Seshadhri (CCC 2009), Kayal & Saraf (FOCS 2009), Saxena & Seshadhri (FOCS 2010, STOC 2011).

Tools employed:
Chinese remaindering over local rings, Sylvester-Gallai configurations, incidence geometry, rank bound estimates, combinatorial arguments on matching/coloring etc.

Known results: Constant top fanin depth-3 circuits

Earlier work:

Dvir & Sphilka (STOC 2005), Kayal & Saxena (CCC 2006), Saxena & Seshadhri (CCC 2009), Kayal & Saraf (FOCS 2009), Saxena & Seshadhri (FOCS 2010, STOC 2011).

Tools employed:

Chinese remaindering over local rings, Sylvester-Gallai configurations, incidence geometry, rank bound estimates, combinatorial arguments on matching/coloring etc.

Known results: Constant read, multilinear depth-4 circuits

Earlier work...

Sphilka & Volkovich (STOC 2008, APPROX-RANDOM 2009), Karnin et al. (STOC 2010), Saraf & Volkovich (STOC 2011), Anderson et al. (CCC 2011).

Tool employed:

Deep structural results on multilinear constant-read circuits.

These results depend very crucially upon multilinearity!

Known results: Constant read, multilinear depth-4 circuits

Earlier work...

Sphilka & Volkovich (STOC 2008, APPROX-RANDOM 2009), Karnin et al. (STOC 2010), Saraf & Volkovich (STOC 2011), Anderson et al. (CCC 2011).

Tool employed:

Deep structural results on multilinear constant-read circuits.

These results depend very crucially upon multilinearity!

A single technique?

Could there be a single tool to handle these two restricted models?

This talk is about one such tool...

Algebraic independence and the Jacobian. (over fields of zero or large characteristic)

A single technique?

Could there be a single tool to handle these two restricted models?

This talk is about one such tool...

Algebraic independence and the Jacobian. (over fields of zero or large characteristic)

Summary of the results

Hitting sets

- We present a single, **common tool** to strictly subsume **all** known cases of **poly-time** hitting sets that have been hitherto solved using diverse tools and techniques (over fields of zero or large characteristic).
- Our work **significantly generalizes** the results obtained by Saxena & Seshadhri (STOC 2011), Saraf & Volkovich (STOC 2011), Anderson et al. (CCC 2011) and Beecken et al. (ICALP 2011), and further brings them under one unifying technique.

Summary of the results

Hitting sets

- We present a single, **common tool** to strictly subsume **all** known cases of **poly-time** hitting sets that have been hitherto solved using diverse tools and techniques (over fields of zero or large characteristic).

- Our work **significantly generalizes** the results obtained by Saxena & Seshadhri (STOC 2011), Saraf & Volkovich (STOC 2011), Anderson et al. (CCC 2011) and Beecken et al. (ICALP 2011), and further brings them under one unifying technique.

Can we prove lower bounds using the Jacobian?

Because of the ‘equivalence’ between identity testing and lower bounds, one might wonder if the Jacobian can also be useful in proving lower bounds.

Summary of the results (contd.)

Lower bounds

- Using the same Jacobian based approach, we prove **exponential lower bounds** for the immanant polynomial on the *same* depth-3 and depth-4 models for which we give hitting sets.

Earlier work on these models did not prove any lower bound results.

Outline

Arithmetic circuits & Identity testing: A brief overview

Algebraic independence and the Jacobian

Hitting sets & lower bounds

Algebraic independence

Algebraic independence:

A set of polynomials $\mathbf{f} = \{f_1, \cdots, f_m\} \subset \mathbb{F}[x_1, \cdots, x_n]$ is **algebraically independent** over $\mathbb{F}$ if there is no non-zero polynomial $H \in \mathbb{F}[y_1, \cdots, y_m]$ such that $H(f_1, \cdots, f_m)$ is identically zero.

A simple example:

Let $f_1 = x^2 - y^2$, $f_2 = x^2 + y$, $f_3 = x$, and $H(z_1, z_2, z_3) = (z_2 - z_3^2)^2 + (z_1 - z_3^2) \neq 0$. Then, $H(f_1, f_2, f_3) = 0$.

Algebraic independence

Algebraic independence:

A set of polynomials $\mathbf{f} = \{f_1, \cdots, f_m\} \subset \mathbb{F}[x_1, \cdots, x_n]$ is **algebraically independent** over $\mathbb{F}$ if there is no non-zero polynomial $H \in \mathbb{F}[y_1, \cdots, y_m]$ such that $H(f_1, \cdots, f_m)$ is identically zero.

A simple example:

Let $f_1 = x^2 - y^2$, $f_2 = x^2 + y$, $f_3 = x$, and $H(z_1, z_2, z_3) = (z_2 - z_3^2)^2 + (z_1 - z_3^2) \neq 0$. Then, $H(f_1, f_2, f_3) = 0$.

Algebraic independence

$\mathbf{f}$ = a set of polynomials.

Transcendence basis:

A maximal subset of $\mathbf{f}$ that is algebraically independent is a **transcendence basis** or (simply) **basis** of $\mathbf{f}$.

Transcendence degree:

The size of such a basis is the **transcendence degree** or **algebraic rank** of $\mathbf{f}$ (denoted by $\mathrm{rk}_{\mathbb{F}}\,\mathbf{f}$). (It is well-defined, and $\mathrm{rk}_{\mathbb{F}}\,\mathbf{f} \leq m$.)

Algebraic independence satisfies the matroid properties.

Algebraic independence

$\mathbf{f}$ = a set of polynomials.

Transcendence basis:

A maximal subset of $\mathbf{f}$ that is algebraically independent is a **transcendence basis** or (simply) **basis** of $\mathbf{f}$.

Transcendence degree:

The size of such a basis is the **transcendence degree** or **algebraic rank** of $\mathbf{f}$ (denoted by $\text{rk}_{\mathbb{F}}\,\mathbf{f}$). (It is well-defined, and $\text{rk}_{\mathbb{F}}\,\mathbf{f} \leq m$.)

Algebraic independence satisfies the matroid properties.

The Jacobian

- The **Jacobian** of polynomials $\mathbf{f} = \{f_1, \cdots, f_m\}$ in $\mathbb{F}[x_1, \cdots, x_n]$ is the matrix,

$$\mathcal{J}_{\mathbf{x}}(\mathbf{f}) = \begin{pmatrix} \partial_{x_1} f_1 & \cdots & \partial_{x_n} f_1 \\ \vdots & \ddots & \vdots \\ \partial_{x_1} f_m & \cdots & \partial_{x_n} f_m \end{pmatrix}_{m \times n}$$

$$\partial_{x_j} f_i := \frac{\partial f_i}{\partial x_j}$$

Jacobian captures algebraic independence

Theorem:

Let $\mathbf{f}$ be a set of polynomials of degree at most d, and $\mathrm{rk}_{\mathbb{F}}\,\mathbf{f} \leq r$. If $\mathrm{char}(\mathbb{F}) = 0$ or $> d^r$ then

$$\mathrm{rk}_{\mathbb{F}}\,\mathbf{f} = \mathrm{rank}_{\mathbb{F}(\mathbf{x})}\mathcal{J}_{\mathbf{x}}(\mathbf{f}).$$

$$\mathbb{F}(\mathbf{x}) = \text{function field on } \mathbf{x}.$$

Naturally,

$$\mathrm{rk}_{\mathbb{F}}\,\mathbf{f} \leq n \quad \text{and} \quad \mathrm{rk}_{\mathbb{F}}\,\mathbf{f} \leq m.$$

Jacobian captures algebraic independence

Theorem:

Let $\mathbf{f}$ be a set of polynomials of degree at most d, and $\mathrm{rk}_{\mathbb{F}}\, \mathbf{f} \leq r$. If $\mathrm{char}(\mathbb{F}) = 0$ or $> d^r$ then

$$\mathrm{rk}_{\mathbb{F}}\, \mathbf{f} = \mathrm{rank}_{\mathbb{F}(\mathbf{x})} \mathcal{J}_{\mathbf{x}}(\mathbf{f}).$$

$\mathbb{F}(\mathbf{x})$ = function field on $\mathbf{x}$.

Naturally,

$$\mathrm{rk}_{\mathbb{F}}\, \mathbf{f} \leq n \quad \text{and} \quad \mathrm{rk}_{\mathbb{F}}\, \mathbf{f} \leq m.$$

Variable reduction: The essence of algebraic rank

"Variable reduction"

In a way...

$\mathrm{rk}_{\mathbb{F}}\,\mathbf{f}$ is a measure of the number of **'hidden'** or **effective** variables in $\mathbf{f}$.

More precisely...

If $\mathrm{rk}_{\mathbb{F}}\,\mathbf{f} = r$ then there exists a **faithful map**,

$$\Phi : x_i \mapsto a_{i1}y_1 + \ldots + a_{ir}y_r + a_{i0}, \qquad a_{ij} \in \mathbb{F}$$

such that $\mathrm{rk}_{\mathbb{F}}\,\mathbf{f} = \mathrm{rk}_{\mathbb{F}}\,\Phi(\mathbf{f}) = r$. $\quad (\Phi(\mathbf{f}) \subset \mathbb{F}[y_1, \ldots, y_r])$

Variable reduction: The essence of algebraic rank

"Variable reduction"

In a way...

$\mathrm{rk}_{\mathbb{F}}\,\mathbf{f}$ is a measure of the number of **'hidden'** or **effective** variables in $\mathbf{f}$.

More precisely...

If $\mathrm{rk}_{\mathbb{F}}\,\mathbf{f} = r$ then there exists a **faithful map**,

$$\Phi : x_i \mapsto a_{i1}y_1 + \ldots + a_{ir}y_r + a_{i0}, \qquad a_{ij} \in \mathbb{F}$$

such that $\mathrm{rk}_{\mathbb{F}}\,\mathbf{f} = \mathrm{rk}_{\mathbb{F}}\,\boldsymbol{\Phi}(\mathbf{f}) = r$. $\qquad (\Phi(\mathbf{f}) \subset \mathbb{F}[y_1, \ldots, y_r])$

Faithful maps preserve non-zeroness

Zero-preserving variable reduction:

If Φ is faithful to $\mathbf{f} = \{f_1, \ldots, f_m\}$ and $C \in \mathbb{F}[z_1, \ldots, z_m]$ then

$$C(\mathbf{f}) = 0 \Leftrightarrow C(\Phi(\mathbf{f})) = 0.$$

Applying algebraic independence to circuits

Let's take the example of depth-3 circuits.

A depth-3 circuit with top fanin m:

$C(f_1, \ldots, f_m) = f_1 + \ldots + f_m$, where f_i is a product of linear polynomials.

- Naturally, $\mathrm{rk}_{\mathbb{F}}\, \mathbf{f} \leq m$.
- Hence, there exists a Φ that reduces the number of variables to less than m, while preserving the 'zero-ness' of C.
- If m is a constant, we can apply 'sparse polynomial PIT' to $\Phi(C)$.

Can we construct Φ efficiently?

Applying algebraic independence to circuits

Let's take the example of depth-3 circuits.

A depth-3 circuit with top fanin m:

$C(f_1, \ldots, f_m) = f_1 + \ldots + f_m$, where f_i is a product of linear polynomials.

- Naturally, $\mathrm{rk}_{\mathbb{F}}\, \mathbf{f} \leq m$.
- Hence, there exists a Φ that reduces the number of variables to less than m, while preserving the 'zero-ness' of C.
- If m is a constant, we can apply 'sparse polynomial PIT' to $\Phi(C)$.

Can we construct Φ efficiently?

Applying algebraic independence to circuits

Let's take the example of depth-3 circuits.

A depth-3 circuit with top fanin m:

$C(f_1, \ldots, f_m) = f_1 + \ldots + f_m$, where f_i is a product of linear polynomials.

- Naturally, $\mathrm{rk}_{\mathbb{F}}\, \mathbf{f} \leq m$.
- Hence, there exists a Φ that reduces the number of variables to less than m, while preserving the 'zero-ness' of C.
- If m is a constant, we can apply 'sparse polynomial PIT' to $\Phi(C)$.

Can we construct Φ efficiently?

Applying algebraic independence to circuits

Let's take the example of depth-3 circuits.

A depth-3 circuit with top fanin m:

$C(f_1, \ldots, f_m) = f_1 + \ldots + f_m$, where f_i is a product of linear polynomials.

- Naturally, $\mathrm{rk}_{\mathbb{F}}\, \mathbf{f} \leq m$.
- Hence, there exists a Φ that reduces the number of variables to less than m, while preserving the 'zero-ness' of C.
- If m is a constant, we can apply 'sparse polynomial PIT' to $\Phi(C)$.

Can we construct Φ efficiently?

Applying algebraic independence to circuits

Let's take the example of depth-3 circuits.

A depth-3 circuit with top fanin m:

$C(f_1, \ldots, f_m) = f_1 + \ldots + f_m$, where f_i is a product of linear polynomials.

- Naturally, $\mathrm{rk}_{\mathbb{F}}\, \mathbf{f} \leq m$.
- Hence, there exists a Φ that reduces the number of variables to less than m, while preserving the 'zero-ness' of C.
- If m is a constant, we can apply 'sparse polynomial PIT' to $\Phi(C)$.

Can we construct Φ efficiently?

Our results: Hitting sets

Result 1: Depth-3 constant top fanin (and more)

Let $C(y_1, \ldots, y_m)$ be any (poly-degree) circuit of size s and each of $f_1, \ldots, f_m$ be a **product of d linear polynomials** in $\mathbb{F}[x_1, \ldots, x_n]$.

If $\mathrm{rk}_{\mathbb{F}} \{f_1, \ldots, f_m\} \leq r$ then a hitting set generator for $C(f_1, \ldots, f_m)$ can be constructed in time $\mathrm{poly}(n, (sd)^r)$.

$\mathrm{char}(\mathbb{F}) = 0$, or $> d^r$.

Corollary: constant top fanin depth-3 circuits:

$C(f_1, \ldots, f_m) = f_1 + \ldots + f_m$ and m is a constant.

Our results: Hitting sets

Result 1: Depth-3 constant top fanin (and more)

Let $C(y_1, \ldots, y_m)$ be any (poly-degree) circuit of size s and each of $f_1, \ldots, f_m$ be a **product of d linear polynomials** in $\mathbb{F}[x_1, \ldots, x_n]$.

If $\text{rk}_{\mathbb{F}} \{f_1, \ldots, f_m\} \leq r$ then a hitting set generator for $C(f_1, \ldots, f_m)$ can be constructed in time $\text{poly}(n, (sd)^r)$.

$$\text{char}(\mathbb{F}) = 0, \text{or} > d^r.$$

Corollary: constant top fanin depth-3 circuits:

$C(f_1, \ldots, f_m) = f_1 + \ldots + f_m$ and m is a constant.

Our results: Hitting sets

Result 1: Depth-3 constant top fanin (and more)

Let $C(y_1, \ldots, y_m)$ be any (poly-degree) circuit of size s and each of $f_1, \ldots, f_m$ be a **product of d linear polynomials** in $\mathbb{F}[x_1, \ldots, x_n]$.

If $\text{rk}_{\mathbb{F}} \{f_1, \ldots, f_m\} \leq r$ then a hitting set generator for $C(f_1, \ldots, f_m)$ can be constructed in time $\text{poly}(n, (sd)^r)$.

$$\text{char}(\mathbb{F}) = 0, \text{or} > d^r.$$

Corollary: constant top fanin depth-3 circuits:

$C(f_1, \ldots, f_m) = f_1 + \ldots + f_m$ and m is a constant.

Occur-k formula

Definition: depth-4 occur-k formula

Let $C = \sum_{i=1}^{m} \prod_{j=1}^{d} P_{ij}^{e_{ij}}$, where P_{ij}'s are sparse polynomials, be a depth-4 circuit.

C is called an **occur**-k depth-4 formula if every variable occurs in at most k of the sparse polynomials P_{ij}'s.

Note: "Constant occur" is a more general concept than "constant read". (Inside a P_{ij} a variable can occur any number of times.) Also, top fan-in m need not be a constant.

Occur-k formula

Definition: depth-4 occur-k formula

Let $C = \sum_{i=1}^{m} \prod_{j=1}^{d} P_{ij}^{e_{ij}}$, where P_{ij}'s are sparse polynomials, be a depth-4 circuit.

C is called an **occur**-k depth-4 formula if every variable occurs in at most k of the sparse polynomials P_{ij}'s.

Note: "Constant occur" is a more general concept than "constant read". (Inside a P_{ij} a variable can occur any number of times.) Also, top fan-in m need not be a constant.

Occur-k formula

Definition: depth-4 occur-k formula

Let $C = \sum_{i=1}^{m} \prod_{j=1}^{d} P_{ij}^{e_{ij}}$, where P_{ij}'s are sparse polynomials, be a depth-4 circuit.

C is called an **occur**-k depth-4 formula if every variable occurs in at most k of the sparse polynomials P_{ij}'s.

Note: "Constant occur" is a more general concept than "constant read". (Inside a P_{ij} a variable can occur any number of times.) Also, top fan-in m need not be a constant.

Strength of 'constant occur'

Kalorkoti (1985):

Constant read formulas cannot express determinant/permanent.

The determinant and permanent polynomials can be computed by an **occur**-1 formula of exponential size - just take the sparse (sum of monomials) representations.

The lower bound question makes sense for occur-const. formula.

Strength of 'constant occur'

Kalorkoti (1985):

Constant read formulas cannot express determinant/permanent.

The determinant and permanent polynomials can be computed by an **occur**-1 formula of exponential size - just take the sparse (sum of monomials) representations.

The lower bound question makes sense for occur-const. formula.

Our results: Hitting sets

Result 2: Depth-4 occur-k

A hitting set generator for a depth-4, occur-k formula of size s can be constructed in time $s^{O(k^2)}$. $(\mathrm{char}(\mathbb{F}) = 0, \text{or} > s^{4k})$

We do not need any restriction of multilinearity on the circuit!

Our results: Hitting sets

Result 2: Depth-4 occur-k

A hitting set generator for a depth-4, occur-k formula of size s can be constructed in time $s^{O(k^2)}$. $(\text{char}(\mathbb{F}) = 0, \text{or} > s^{4k})$

We do not need any restriction of multilinearity on the circuit!

Our results: Hitting sets

Result 3: Depth-D occur-k

A hitting set generator for a depth-D, occur-k formula of size s can be constructed in time polynomial in s^R, where $R = (2k)^{2D \cdot 2^D}$. $(\mathrm{char}(\mathbb{F}) = 0, \text{or} > s^R)$

Once again, no multilinearity assumption!

Our results: Hitting sets

Result 3: Depth-D occur-k

A hitting set generator for a depth-D, occur-k formula of size s can be constructed in time polynomial in s^R, where $R = (2k)^{2D \cdot 2^D}$. $(\mathrm{char}(\mathbb{F}) = 0, \text{or} > s^R)$

Once again, no multilinearity assumption!

Towards a lower bound: Immanant polynomial

Definition:

For any character $\chi : S_n \rightarrow \mathbb{C}^\times$, the **immanant** of a matrix $M = (x_{ij})_{n\times n}$ with respect to χ is defined as

$$\mathrm{Imm}_n = \mathrm{Imm}_\chi(M) = \sum_{\sigma \in S_n} \chi(\sigma) \prod_{i=1}^{n} x_{i\sigma(i)}.$$

Determinant & permanent are special cases of the immanant.

Lower bound

Result 4: Depth-4 occur-k

Any depth-4 occur-k formula that computes Imm_n must have size $s = 2^{\Omega(n/k^2)}$. $(\mathrm{char}(\mathbb{F}) = 0)$

Corollary:

If every variable occurs in at most $n^{1/2-\epsilon}$ $(0 < \epsilon < 1/2)$ many 'underlying' sparse polynomials, then it takes a $2^{\Omega(n^{2\epsilon})}$-sized depth-4 circuits to compute Imm_n.

Ideally, we would like to allow poly(n)-occurrence of a variable and get a $2^{\Omega(n)}$ lower bound for depth-4 circuits, in order to show that VP $\neq$ VNP.

Lower bound

Result 4: Depth-4 occur-k

Any depth-4 occur-k formula that computes Imm_n must have size $s = 2^{\Omega(n/k^2)}$. $(\mathrm{char}(\mathbb{F}) = 0)$

Corollary:

If every variable occurs in at most $n^{1/2-\epsilon}$ $(0 < \epsilon < 1/2)$ many 'underlying' sparse polynomials, then it takes a $2^{\Omega(n^{2\epsilon})}$-sized depth-4 circuits to compute Imm_n.

Ideally, we would like to allow poly(n)-occurrence of a variable and get a $2^{\Omega(n)}$ lower bound for depth-4 circuits, in order to show that VP $\neq$ VNP.

Lower bound

Result 4: Depth-4 occur-k

Any depth-4 occur-k formula that computes Imm_n must have size $s = 2^{\Omega(n/k^2)}$. $(\mathrm{char}(\mathbb{F}) = 0)$

Corollary:

If every variable occurs in at most $n^{1/2-\epsilon}$ $(0 < \epsilon < 1/2)$ many 'underlying' sparse polynomials, then it takes a $2^{\Omega(n^{2\epsilon})}$-sized depth-4 circuits to compute Imm_n.

Ideally, we would like to allow poly(n)-occurrence of a variable and get a $2^{\Omega(n)}$ lower bound for depth-4 circuits, in order to show that VP $\neq$ VNP.

Open question

Depth-4, top fanin-m circuit:

$C(f_1, \ldots, f_m) = f_1 + \ldots + f_m$, where f_i is a product of sparse polynomials.

Open question:

Can we efficiently compute a faithful map Φ for C when m is a constant?

- Such a Φ exists.
- We could compute Φ efficiently for the 'depth-3 const-top fanin' and the 'depth-4, occur-k' models using the Jacobian.

Open question

Depth-4, top fanin-m circuit:

$C(f_1, \ldots, f_m) = f_1 + \ldots + f_m$, where f_i is a product of sparse polynomials.

Open question:

Can we efficiently compute a faithful map Φ for C when m is a constant?

- Such a Φ exists.
- We could compute Φ efficiently for the 'depth-3 const-top fanin' and the 'depth-4, occur-k' models using the Jacobian.

Open question

Depth-4, top fanin-m circuit:

$C(f_1, \ldots, f_m) = f_1 + \ldots + f_m$, where f_i is a product of sparse polynomials.

Open question:

Can we efficiently compute a faithful map Φ for C when m is a constant?

- Such a Φ exists.
- We could compute Φ efficiently for the 'depth-3 const-top fanin' and the 'depth-4, occur-k' models using the Jacobian.

Open question

Depth-4, top fanin-m circuit:

$C(f_1, \ldots, f_m) = f_1 + \ldots + f_m$, where f_i is a product of sparse polynomials.

Open question:

Can we efficiently compute a faithful map Φ for C when m is a constant?

- Such a Φ exists.
- We could compute Φ efficiently for the 'depth-3 const-top fanin' and the 'depth-4, occur-k' models using the Jacobian.

Proof ideas

Outline

Proof outline of the following results

- Result 2: (hitting set) A hitting set generator for a depth-4, occur-k formula of size s can be constructed in time $s^{O(k^2)}$.
- Result 4: (lower bound) Any depth-4 occur-k formula that computes Imm_n must have size $s = 2^{\Omega(n/k^2)}$.

Assume, $\text{char}(\mathbb{F}) = 0$.

Proof outline of the following results

- Result 2: (hitting set) A hitting set generator for a depth-4, occur-k formula of size s can be constructed in time $s^{O(k^2)}$.
- Result 4: (lower bound) Any depth-4 occur-k formula that computes Imm_n must have size $s = 2^{\Omega(n/k^2)}$.

Assume, $\mathrm{char}(\mathbb{F}) = 0$.

Proof outline of the following results

- Result 2: (hitting set) A hitting set generator for a depth-4, occur-k formula of size s can be constructed in time $s^{O(k^2)}$.
- Result 4: (lower bound) Any depth-4 occur-k formula that computes Imm_n must have size $s = 2^{\Omega(n/k^2)}$.

Assume, $\text{char}(\mathbb{F}) = 0$.

Constructing faithful maps

Definition: Faithful homomorphism

A homomorphism $\Phi : \mathbb{F}[\mathbf{x}] \to \mathbb{F}[y_1, \ldots, y_k]$ is said to be **faithful** to a set of polynomials $\mathbf{f} \subset \mathbb{F}[\mathbf{x}]$ if $\mathrm{rk}_{\mathbb{F}}\, \mathbf{f} = \mathrm{rk}_{\mathbb{F}}\, \Phi(\mathbf{f}) = r$.

Lemma: Chain rule on Jacobian

If $\Phi : \mathbb{F}[\mathbf{x}] \to \mathbb{F}[y_1, \ldots, y_k]$ is a homomorphism then

$$\underbrace{\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))}_{m \times k} = \underbrace{\Phi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))}_{m \times n} \cdot \underbrace{\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{x}))}_{n \times k}.$$

Constructing faithful maps

Definition: Faithful homomorphism

A homomorphism $\Phi : \mathbb{F}[\mathbf{x}] \to \mathbb{F}[y_1, \ldots, y_k]$ is said to be **faithful** to a set of polynomials $\mathbf{f} \subset \mathbb{F}[\mathbf{x}]$ if $\mathrm{rk}_{\mathbb{F}}\, \mathbf{f} = \mathrm{rk}_{\mathbb{F}}\, \Phi(\mathbf{f}) = r$.

Lemma: Chain rule on Jacobian

If $\Phi : \mathbb{F}[\mathbf{x}] \to \mathbb{F}[y_1, \ldots, y_k]$ is a homomorphism then

$$\underbrace{\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))}_{m \times k} = \underbrace{\Phi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))}_{m \times n} \cdot \underbrace{\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{x}))}_{n \times k}.$$

Constructing faithful maps

Proof: Chain rule on Jacobian

Let $f_i = \sum_j c_j \cdot \mathbf{m_j}$, where $\mathbf{m_j} = x_1^{e_{j1}} \cdots x_n^{e_{jn}}$ and $c_j \in \mathbb{F}$. Then,

$$
\begin{aligned}
\Phi(f_i) &= \sum_j c_j \cdot \Phi(x_i)^{e_{j1}} \cdots \Phi(x_n)^{e_{jn}} \\
\Rightarrow \partial_y(\Phi(f_i)) &= \sum_j c_j \cdot \sum_{k=1}^{n} e_{j\ell} \frac{\Phi(\mathbf{m_j})}{\Phi(x_\ell)} \cdot \frac{\partial \Phi(x_\ell)}{\partial y} \\
&= \sum_{\ell=1}^{n} \left(\sum_j c_j \cdot e_{j\ell} \frac{\Phi(\mathbf{m_j})}{\Phi(x_\ell)} \right) \cdot \frac{\partial \Phi(x_\ell)}{\partial y} \\
&= \sum_{\ell=1}^{n} \Phi\left(\frac{\partial f_i}{\partial x_\ell} \right) \cdot \frac{\partial \Phi(x_\ell)}{\partial y}
\end{aligned}
$$

Constructing faithful maps

Proof: Chain rule on Jacobian

Let $f_i = \sum_j c_j \cdot \mathbf{m_j}$, where $\mathbf{m_j} = x_1^{e_{j1}} \cdots x_n^{e_{jn}}$ and $c_j \in \mathbb{F}$. Then,

$$\Phi(f_i) = \sum_j c_j \cdot \Phi(x_i)^{e_{j1}} \cdots \Phi(x_n)^{e_{jn}}$$

$$\Rightarrow \partial_y(\Phi(f_i)) = \sum_j c_j \cdot \sum_{k=1}^{n} e_{j\ell} \frac{\Phi(\mathbf{m_j})}{\Phi(x_\ell)} \cdot \frac{\partial \Phi(x_\ell)}{\partial y}$$

$$= \sum_{\ell=1}^{n} \left(\sum_j c_j \cdot e_{j\ell} \frac{\Phi(\mathbf{m_j})}{\Phi(x_\ell)} \right) \cdot \frac{\partial \Phi(x_\ell)}{\partial y}$$

$$= \sum_{\ell=1}^{n} \Phi\left(\frac{\partial f_i}{\partial x_\ell} \right) \cdot \frac{\partial \Phi(x_\ell)}{\partial y}$$

Constructing faithful maps

Proof: Chain rule on Jacobian

Let $f_i = \sum_j c_j \cdot \mathbf{m_j}$, where $\mathbf{m_j} = x_1^{e_{j1}} \cdots x_n^{e_{jn}}$ and $c_j \in \mathbb{F}$. Then,

$$\begin{aligned} \Phi(f_i) &= \sum_j c_j \cdot \Phi(x_i)^{e_{j1}} \cdots \Phi(x_n)^{e_{jn}} \\ \Rightarrow \partial_y(\Phi(f_i)) &= \sum_j c_j \cdot \sum_{k=1}^{n} e_{j\ell} \frac{\Phi(\mathbf{m_j})}{\Phi(x_\ell)} \cdot \frac{\partial \Phi(x_\ell)}{\partial y} \\ &= \sum_{\ell=1}^{n} \left(\sum_j c_j \cdot e_{j\ell} \frac{\Phi(\mathbf{m_j})}{\Phi(x_\ell)} \right) \cdot \frac{\partial \Phi(x_\ell)}{\partial y} \\ &= \sum_{\ell=1}^{n} \Phi\left(\frac{\partial f_i}{\partial x_\ell}\right) \cdot \frac{\partial \Phi(x_\ell)}{\partial y} \end{aligned}$$

Constructing faithful maps

Proof: Chain rule on Jacobian

Let $f_i = \sum_j c_j \cdot \mathbf{m_j}$, where $\mathbf{m_j} = x_1^{e_{j1}} \cdots x_n^{e_{jn}}$ and $c_j \in \mathbb{F}$. Then,

$$
\begin{aligned}
\Phi(f_i) &= \sum_j c_j \cdot \Phi(x_i)^{e_{j1}} \cdots \Phi(x_n)^{e_{jn}} \\
\Rightarrow \partial_y(\Phi(f_i)) &= \sum_j c_j \cdot \sum_{k=1}^{n} e_{j\ell} \frac{\Phi(\mathbf{m_j})}{\Phi(x_\ell)} \cdot \frac{\partial \Phi(x_\ell)}{\partial y} \\
&= \sum_{\ell=1}^{n} \left(\sum_j c_j \cdot e_{j\ell} \frac{\Phi(\mathbf{m_j})}{\Phi(x_\ell)} \right) \cdot \frac{\partial \Phi(x_\ell)}{\partial y} \\
&= \sum_{\ell=1}^{n} \Phi\left(\frac{\partial f_i}{\partial x_\ell} \right) \cdot \frac{\partial \Phi(x_\ell)}{\partial y}
\end{aligned}
$$

Constructing faithful maps

Proof: Chain rule on Jacobian

Let $f_i = \sum_j c_j \cdot \mathbf{m_j}$, where $\mathbf{m_j} = x_1^{e_{j1}} \cdots x_n^{e_{jn}}$ and $c_j \in \mathbb{F}$. Then,

$$\Phi(f_i) = \sum_j c_j \cdot \Phi(x_i)^{e_{j1}} \cdots \Phi(x_n)^{e_{jn}}$$

$$\Rightarrow \partial_y(\Phi(f_i)) = \sum_j c_j \cdot \sum_{k=1}^{n} e_{j\ell} \frac{\Phi(\mathbf{m_j})}{\Phi(x_\ell)} \cdot \frac{\partial \Phi(x_\ell)}{\partial y}$$

$$= \sum_{\ell=1}^{n} \left(\sum_j c_j \cdot e_{j\ell} \frac{\Phi(\mathbf{m_j})}{\Phi(x_\ell)} \right) \cdot \frac{\partial \Phi(x_\ell)}{\partial y}$$

$$= \sum_{\ell=1}^{n} \Phi\left(\frac{\partial f_i}{\partial x_\ell}\right) \cdot \frac{\partial \Phi(x_\ell)}{\partial y}$$

Constructing faithful maps

Proof: Chain rule on Jacobian

Let $f_i = \sum_j c_j \cdot \mathbf{m_j}$, where $\mathbf{m_j} = x_1^{e_{j1}} \cdots x_n^{e_{jn}}$ and $c_j \in \mathbb{F}$. Then,

$$\begin{aligned}
\Phi(f_i) &= \sum_j c_j \cdot \Phi(x_i)^{e_{j1}} \cdots \Phi(x_n)^{e_{jn}} \\
\Rightarrow \partial_y(\Phi(f_i)) &= \sum_j c_j \cdot \sum_{k=1}^{n} e_{j\ell} \frac{\Phi(\mathbf{m_j})}{\Phi(x_\ell)} \cdot \frac{\partial \Phi(x_\ell)}{\partial y} \\
&= \sum_{\ell=1}^{n} \left(\sum_j c_j \cdot e_{j\ell} \frac{\Phi(\mathbf{m_j})}{\Phi(x_\ell)} \right) \cdot \frac{\partial \Phi(x_\ell)}{\partial y} \\
&= \sum_{\ell=1}^{n} \Phi\left(\frac{\partial f_i}{\partial x_\ell} \right) \cdot \frac{\partial \Phi(x_\ell)}{\partial y}
\end{aligned}$$

Constructing faithful maps

$$\underbrace{\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))}_{m\times k} = \underbrace{\Phi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))}_{m\times n} \cdot \underbrace{\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{x}))}_{n\times k}$$

We would like to make $\text{rank}_{\mathbb{F}(\mathbf{y})}\left[\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))\right] = \text{rank}_{\mathbb{F}(\mathbf{x})}\left[\mathcal{J}_{\mathbf{x}}(\mathbf{f})\right]$

Theorem:
Let $\text{rk}_{\mathbb{F}}\, \mathbf{f} = r \leq k$, and $\Psi : \mathbb{F}[\mathbf{x}] \to \mathbb{F}[\mathbf{z}]$ be a homomorphism s.t.

$$\text{rank}_{\mathbb{F}(\mathbf{x})}\left[\mathcal{J}_{\mathbf{x}}(\mathbf{f})\right] = \text{rank}_{\mathbb{F}(\mathbf{z})}\left[\Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))\right].$$

Then, the map $\Phi : x_i \to \left(\sum_{j=1}^{k} y_j t^{ij}\right) + \Psi(x_i)$ from $\mathbb{F}[\mathbf{x}]$ to $\mathbb{F}[y_1, \ldots, y_k, t, \mathbf{z}]$ is a homomorphism, faithful to $\mathbf{f}$.

Constructing faithful maps

$$\underbrace{\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))}_{m\times k} = \underbrace{\Phi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))}_{m\times n} \cdot \underbrace{\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{x}))}_{n\times k}$$

We would like to make $\mathrm{rank}_{\mathbb{F}(\mathbf{y})}\,[\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))] = \mathrm{rank}_{\mathbb{F}(\mathbf{x})}\,[\mathcal{J}_{\mathbf{x}}(\mathbf{f})]$

Theorem:
Let $\mathrm{rk}_{\mathbb{F}}\,\mathbf{f} = r \leq k$, and $\Psi : \mathbb{F}[\mathbf{x}] \to \mathbb{F}[\mathbf{z}]$ be a homomorphism s.t.

$$\mathrm{rank}_{\mathbb{F}(\mathbf{x})}\,[\mathcal{J}_{\mathbf{x}}(\mathbf{f})] = \mathrm{rank}_{\mathbb{F}(\mathbf{z})}\,[\Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))].$$

Then, the map $\Phi : x_i \to \left(\sum_{j=1}^{k} y_j t^{ij}\right) + \Psi(x_i)$ from $\mathbb{F}[\mathbf{x}]$ to $\mathbb{F}[y_1, \ldots, y_k, t, \mathbf{z}]$ is a homomorphism, faithful to $\mathbf{f}$.

Constructing faithful maps

$$\underbrace{\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))}_{m\times k} = \underbrace{\Phi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))}_{m\times n} \cdot \underbrace{\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{x}))}_{n\times k}$$

We would like to make $\mathrm{rank}_{\mathbb{F}(\mathbf{y})}\,[\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))] = \mathrm{rank}_{\mathbb{F}(\mathbf{x})}\,[\mathcal{J}_{\mathbf{x}}(\mathbf{f})]$

Theorem:
Let $\mathrm{rk}_{\mathbb{F}}\,\mathbf{f} = r \leq k$, and $\Psi : \mathbb{F}[\mathbf{x}] \to \mathbb{F}[\mathbf{z}]$ be a homomorphism s.t.

$$\mathrm{rank}_{\mathbb{F}(\mathbf{x})}\,[\mathcal{J}_{\mathbf{x}}(\mathbf{f})] = \mathrm{rank}_{\mathbb{F}(\mathbf{z})}\,[\Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))].$$

Then, the map $\Phi : x_i \to \left(\sum_{j=1}^{k} y_j t^{ij}\right) + \Psi(x_i)$ from $\mathbb{F}[\mathbf{x}]$ to $\mathbb{F}[y_1, \ldots, y_k, t, \mathbf{z}]$ is a homomorphism, faithful to $\mathbf{f}$.

Constructing faithful maps

Wait a min...

We promised that Φ is a map from $\mathbb{F}[\mathbf{x}]$ to $\mathbb{F}[y_1, \ldots, y_k]$ - what are these extra variables $t, \mathbf{z}$ doing here?

Refining Φ

Pretend that $\Phi : \mathbb{F}[\mathbf{x}] \mapsto \mathbb{F}(t, \mathbf{z})[y_1, \ldots, y_k]$.

Note that even with this 'new' Φ

$$\underbrace{\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))}_{m \times k} = \underbrace{\Phi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))}_{m \times n} \cdot \underbrace{\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{x}))}_{n \times k}$$

Refined goal

To show that $\text{rank}_{\mathbb{F}(t,\mathbf{z},\mathbf{y})} [\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))] = \text{rank}_{\mathbb{F}(\mathbf{x})} [\mathcal{J}_{\mathbf{x}}(\mathbf{f})]$

Constructing faithful maps

Wait a min...

We promised that Φ is a map from $\mathbb{F}[\mathbf{x}]$ to $\mathbb{F}[y_1, \ldots, y_k]$ - what are these extra variables $t, \mathbf{z}$ doing here?

Refining Φ

Pretend that $\Phi : \mathbb{F}[\mathbf{x}] \mapsto \mathbb{F}(t, \mathbf{z})[y_1, \ldots, y_k]$.

Note that even with this 'new' Φ

$$\underbrace{\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))}_{m \times k} = \underbrace{\Phi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))}_{m \times n} \cdot \underbrace{\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{x}))}_{n \times k}$$

Refined goal

To show that $\text{rank}_{\mathbb{F}(t,\mathbf{z},\mathbf{y})} [\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))] = \text{rank}_{\mathbb{F}(\mathbf{x})} [\mathcal{J}_{\mathbf{x}}(\mathbf{f})]$

Constructing faithful maps

Wait a min...

We promised that Φ is a map from $\mathbb{F}[\mathbf{x}]$ to $\mathbb{F}[y_1, \ldots, y_k]$ - what are these extra variables $t, \mathbf{z}$ doing here?

Refining Φ

Pretend that $\Phi : \mathbb{F}[\mathbf{x}] \mapsto \mathbb{F}(t, \mathbf{z})[y_1, \ldots, y_k]$.

Note that even with this 'new' Φ

$$\underbrace{\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))}_{m \times k} = \underbrace{\Phi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))}_{m \times n} \cdot \underbrace{\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{x}))}_{n \times k}$$

Refined goal

To show that $\operatorname{rank}_{\mathbb{F}(t,\mathbf{z},\mathbf{y})} [\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))] = \operatorname{rank}_{\mathbb{F}(\mathbf{x})} [\mathcal{J}_{\mathbf{x}}(\mathbf{f})]$

Constructing faithful maps

Wait a min...

We promised that Φ is a map from $\mathbb{F}[\mathbf{x}]$ to $\mathbb{F}[y_1, \ldots, y_k]$ - what are these extra variables $t, \mathbf{z}$ doing here?

Refining Φ

Pretend that $\Phi : \mathbb{F}[\mathbf{x}] \mapsto \mathbb{F}(t, \mathbf{z})[y_1, \ldots, y_k]$.

Note that even with this 'new' Φ

$$\underbrace{\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))}_{m \times k} = \underbrace{\Phi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))}_{m \times n} \cdot \underbrace{\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{x}))}_{n \times k}$$

Refined goal

To show that $\text{rank}_{\mathbb{F}(t,\mathbf{z},\mathbf{y})} \left[\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))\right] = \text{rank}_{\mathbb{F}(\mathbf{x})} \left[\mathcal{J}_{\mathbf{x}}(\mathbf{f})\right]$

Constructing faithful maps

$$\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f})) = \Phi(\mathcal{J}_{\mathbf{x}}(\mathbf{f})) \cdot \mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{x})) \quad \text{and}$$

$$\Phi : x_i \to \left(\sum_{j=1}^{k} y_j t^{ij}\right) + \Psi(x_i)$$

Proof:

$$\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f})) = \Phi(\mathcal{J}_{\mathbf{x}}(\mathbf{f})) \cdot \begin{pmatrix} \cdots & t^{j} \cdots \\ \cdots & t^{2j} \cdots \\ & \vdots \\ \cdots & \vdots \cdots \\ \cdots & t^{nj} \cdots \end{pmatrix}_{n \times k}.$$

Hence at $\mathbf{y} = 0$,

$$\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f})) = \Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f})) \cdot \begin{pmatrix} \cdots & t^{j} \cdots \\ \cdots & t^{2j} \cdots \\ & \vdots \\ \cdots & \vdots \cdots \\ \cdots & t^{nj} \cdots \end{pmatrix}.$$

Note: $\Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))$ is t-free.

Constructing faithful maps

$$\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f})) = \Phi(\mathcal{J}_{\mathbf{x}}(\mathbf{f})) \cdot \mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{x})) \quad \text{and}$$

$$\Phi : x_i \to \left(\sum_{j=1}^{k} y_j t^{ij}\right) + \Psi(x_i)$$

Proof:

$$\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f})) = \Phi(\mathcal{J}_{\mathbf{x}}(\mathbf{f})) \cdot \begin{pmatrix} \dots & t^{j} \dots \\ \dots & t^{2j} \dots \\ \dots & \vdots \dots \\ \dots & t^{nj} \dots \end{pmatrix}_{n \times k}.$$

Hence at $\mathbf{y} = 0$,

$$\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f})) = \Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f})) \cdot \begin{pmatrix} \dots & t^{j} \dots \\ \dots & t^{2j} \dots \\ \dots & \vdots \dots \\ \dots & t^{nj} \dots \end{pmatrix}.$$

Note: $\Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))$ is t-free.

Constructing faithful maps

$$\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f})) = \Phi(\mathcal{J}_{\mathbf{x}}(\mathbf{f})) \cdot \mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{x})) \quad \text{and}$$

$$\Phi : x_i \to \left(\sum_{j=1}^{k} y_j t^{ij}\right) + \Psi(x_i)$$

Proof:

$$\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f})) = \Phi(\mathcal{J}_{\mathbf{x}}(\mathbf{f})) \cdot \begin{pmatrix} \dots & t^j \dots \\ \dots & t^{2j} \dots \\ \dots & \vdots \dots \\ \dots & t^{nj} \dots \end{pmatrix}_{n \times k}. \text{ Hence at } \mathbf{y} = 0,$$

$$\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f})) = \Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f})) \cdot \begin{pmatrix} \dots & t^j \dots \\ \dots & t^{2j} \dots \\ \dots & \vdots \dots \\ \dots & t^{nj} \dots \end{pmatrix}.$$

Note: $\Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))$ is t-free.

Constructing faithful maps

$$\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))_{\mathbf{y}=0} = \Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f})) \cdot \begin{pmatrix} \dots & t^{j} \dots \\ \dots & t^{2j} \dots \\ & \vdots \\ \dots & \vdots \dots \\ \dots & t^{nj} \dots \end{pmatrix}.$$

Proof (contd.):

Therefore, (by the 'Vandermonde nature' of the t-matrix)

$$\text{rank}[\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))] \geq \text{rank}[\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))]_{\mathbf{y}=0} = \text{rank}[\Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))] = \text{rank}[\mathcal{J}_{\mathbf{x}}(\mathbf{f})].$$

Surely, $\text{rank}[\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))] \leq \text{rank}[\mathcal{J}_{\mathbf{x}}(\mathbf{f})]$.($\Phi$ can only decrease alg. rk.)
Hence, $\text{rank}[\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))] = \text{rank}[\mathcal{J}_{\mathbf{x}}(\mathbf{f})]$.

Constructing faithful maps

$$\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))_{\mathbf{y}=0} = \Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f})) \cdot \begin{pmatrix} \dots & t^{j} \dots \\ \dots & t^{2j} \dots \\ \dots & \vdots \dots \\ \dots & t^{nj} \dots \end{pmatrix}.$$

Proof (contd.):

Therefore, (by the 'Vandermonde nature' of the t-matrix)

$$\text{rank}[\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))] \geq \text{rank}[\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))]_{\mathbf{y}=0} = \text{rank}[\Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))] = \text{rank}[\mathcal{J}_{\mathbf{x}}(\mathbf{f})].$$

Surely, $\text{rank}[\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))] \leq \text{rank}[\mathcal{J}_{\mathbf{x}}(\mathbf{f})]$.($\Phi$ can only decrease alg. rk.)
Hence, $\text{rank}[\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))] = \text{rank}[\mathcal{J}_{\mathbf{x}}(\mathbf{f})]$.

Constructing faithful maps

$$\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))_{\mathbf{y}=0} = \Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f})) \cdot \begin{pmatrix} \dots & t^j \dots \\ \dots & t^{2j} \dots \\ \dots & \vdots \dots \\ \dots & t^{nj} \dots \end{pmatrix}.$$

Proof (contd.):

Therefore, (by the 'Vandermonde nature' of the t-matrix)

$$\mathrm{rank}[\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))] \geq \mathrm{rank}[\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))]_{\mathbf{y}=0} = \mathrm{rank}[\Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))] = \mathrm{rank}[\mathcal{J}_{\mathbf{x}}(\mathbf{f})].$$

Surely, $\mathrm{rank}[\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))] \leq \mathrm{rank}[\mathcal{J}_{\mathbf{x}}(\mathbf{f})]$.($\Phi$ can only decrease alg. rk.)
Hence, $\mathrm{rank}[\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))] = \mathrm{rank}[\mathcal{J}_{\mathbf{x}}(\mathbf{f})]$.

Constructing faithful maps

$$\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))_{\mathbf{y}=0} = \Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f})) \cdot \begin{pmatrix} \ldots & t^{j} \ldots \\ \ldots & t^{2j} \ldots \\ \ldots & \vdots \ldots \\ \ldots & t^{nj} \ldots \end{pmatrix}.$$

Proof (contd.):

Therefore, (by the 'Vandermonde nature' of the t-matrix)

$$\text{rank}[\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))] \geq \text{rank}[\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))]_{\mathbf{y}=0} = \text{rank}[\Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))] = \text{rank}[\mathcal{J}_{\mathbf{x}}(\mathbf{f})].$$

Surely, $\text{rank}[\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))] \leq \text{rank}[\mathcal{J}_{\mathbf{x}}(\mathbf{f})]$.($\Phi$ can only decrease alg. rk.)

Hence, $\text{rank}[\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))] = \text{rank}[\mathcal{J}_{\mathbf{x}}(\mathbf{f})]$.

Constructing faithful maps

$$\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))_{\mathbf{y}=0} = \Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f})) \cdot \begin{pmatrix} \ldots & t^{j} \ldots \\ \ldots & t^{2j} \ldots \\ \ldots & \vdots \ldots \\ \ldots & t^{nj} \ldots \end{pmatrix}.$$

Proof (contd.):

Therefore, (by the 'Vandermonde nature' of the t-matrix)

$$\text{rank}[\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))] \geq \text{rank}[\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))]_{\mathbf{y}=0} = \text{rank}[\Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))] = \text{rank}[\mathcal{J}_{\mathbf{x}}(\mathbf{f})].$$

Surely, $\text{rank}[\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))] \leq \text{rank}[\mathcal{J}_{\mathbf{x}}(\mathbf{f})]$.($\Phi$ can only decrease alg. rk.)
Hence, $\text{rank}[\mathcal{J}_{\mathbf{y}}(\Phi(\mathbf{f}))] = \text{rank}[\mathcal{J}_{\mathbf{x}}(\mathbf{f})]$.

Constructing faithful maps

What have we achieved?

The problem boils down to constructing a map Ψ such that

$$\mathrm{rank}_{\mathbb{F}(\mathbf{x})}\left[\mathcal{J}_{\mathbf{x}}(\mathbf{f})\right] = \mathrm{rank}_{\mathbb{F}(\mathbf{z})}\left[\Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))\right].$$

We would want...

Ψ to introduce as few $\mathbf{z}$ variables as possible, or else Φ won't be an efficient map.

Constructing faithful maps

What have we achieved?

The problem boils down to constructing a map Ψ such that

$$\text{rank}_{\mathbb{F}(\mathbf{x})}\left[\mathcal{J}_{\mathbf{x}}(\mathbf{f})\right] = \text{rank}_{\mathbb{F}(\mathbf{z})}\left[\Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))\right].$$

We would want...

Ψ to introduce as few $\mathbf{z}$ variables as possible, or else Φ won't be an efficient map.

How to preserve rank of the Jacobian under Ψ?

This is where the particular model of circuits come in the picture.

Depth-4 circuit: $C = \sum_{i=1}^{m} \prod_{j=1}^{d} P_{ij}$, where P_{ij}'s are sparse polynomials with sparsity bounded by s.

A certain simplification:

If C is an **occur**-k circuit then we can assume that $m \leq 2k$.

Proof:

There exists an x_i s.t. $C = 0 \Leftrightarrow C' := C(x_i + 1) - C(x_i) = 0$.
Circuit C' has top fanin at most $2k$.
It is easy to construct a hitting set for C from that of C'.

How to preserve rank of the Jacobian under Ψ?

This is where the particular model of circuits come in the picture.

Depth-4 circuit: $C = \sum_{i=1}^{m} \prod_{j=1}^{d} P_{ij}$, where P_{ij}'s are sparse polynomials with sparsity bounded by s.

A certain simplification:

If C is an **occur**-k circuit then we can assume that $m \leq 2k$.

Proof:

There exists an x_i s.t. $C = 0 \Leftrightarrow C' := C(x_i + 1) - C(x_i) = 0$.

Circuit C' has top fanin at most $2k$.

It is easy to construct a hitting set for C from that of C'.

How to preserve rank of the Jacobian under Ψ?

This is where the particular model of circuits come in the picture.

Depth-4 circuit: $C = \sum_{i=1}^{m} \prod_{j=1}^{d} P_{ij}$, where P_{ij}'s are sparse polynomials with sparsity bounded by s.

A certain simplification:

If C is an **occur**-k circuit then we can assume that $m \leq 2k$.

Proof:

There exists an x_i s.t. $C = 0 \Leftrightarrow C' := C(x_i + 1) - C(x_i) = 0$.
Circuit C' has top fanin at most $2k$.
It is easy to construct a hitting set for C from that of C'.

How to preserve rank of the Jacobian under Ψ?

Let $f_i = \prod_{j=1}^{d} P_{ij}$. Then $C = \sum_{i=1}^{2k} f_i$. Let, $\mathrm{rank}_{\mathbb{F}(\mathbf{x})}[\mathcal{J}_{\mathbf{x}}(\mathbf{f})] = r$

Claim:

Any $r \times r$ minor of the $\mathcal{J}_{\mathbf{x}}(\mathbf{f})$ can be expressed as a **product of sparse polynomials** with sparsity bounded by $s^{O(k^2)}$.

Proof: Focus on a minor of $\mathcal{J}_{\mathbf{x}}(\mathbf{f})$, let's say

$$\det \begin{pmatrix} \partial_{x_1} f_1 & \dots & \partial_{x_r} f_1 \\ \partial_{x_1} f_2 & \dots & \partial_{x_r} f_2 \\ \vdots & & \\ \partial_{x_1} f_{2k} & \dots & \partial_{x_r} f_{2k} \end{pmatrix}$$

$x_1, \dots, x_r$ occur in at most $2kr$ many P_{ij}'s. So, most of the P_{ij}'s come out common from the determinant.

How to preserve rank of the Jacobian under Ψ?

Let $f_i = \prod_{j=1}^{d} P_{ij}$. Then $C = \sum_{i=1}^{2k} f_i$. Let, $\text{rank}_{\mathbb{F}(\mathbf{x})}[\mathcal{J}_{\mathbf{x}}(\mathbf{f})] = r$

Claim:

Any $r \times r$ minor of the $\mathcal{J}_{\mathbf{x}}(\mathbf{f})$ can be expressed as a **product of sparse polynomials** with sparsity bounded by $s^{O(k^2)}$.

Proof: Focus on a minor of $\mathcal{J}_{\mathbf{x}}(\mathbf{f})$, let's say

$$\det \begin{pmatrix} \partial_{x_1} f_1 & \ldots & \partial_{x_r} f_1 \\ \partial_{x_1} f_2 & \ldots & \partial_{x_r} f_2 \\ \vdots & & \\ \partial_{x_1} f_{2k} & \ldots & \partial_{x_r} f_{2k} \end{pmatrix}$$

$x_1, \ldots, x_r$ occur in at most $2kr$ many P_{ij}'s. So, most of the P_{ij}'s come out common from the determinant.

How to preserve rank of the Jacobian under Ψ?

Let $f_i = \prod_{j=1}^{d} P_{ij}$. Then $C = \sum_{i=1}^{2k} f_i$. Let, $\text{rank}_{\mathbb{F}(\mathbf{x})}[\mathcal{J}_{\mathbf{x}}(\mathbf{f})] = r$

Claim:

Any $r \times r$ minor of the $\mathcal{J}_{\mathbf{x}}(\mathbf{f})$ can be expressed as a **product of sparse polynomials** with sparsity bounded by $s^{O(k^2)}$.

Proof: Focus on a minor of $\mathcal{J}_{\mathbf{x}}(\mathbf{f})$, let's say

$$\det \begin{pmatrix} \partial_{x_1} f_1 & \dots & \partial_{x_r} f_1 \\ \partial_{x_1} f_2 & \dots & \partial_{x_r} f_2 \\ \vdots & & \\ \partial_{x_1} f_{2k} & \dots & \partial_{x_r} f_{2k} \end{pmatrix}$$

$x_1, \dots, x_r$ occur in at most $2kr$ many P_{ij}'s. So, most of the P_{ij}'s come out common from the determinant.

How to preserve rank of the Jacobian under Ψ?

Let $f_i = \prod_{j=1}^{d} P_{ij}$. Then $C = \sum_{i=1}^{2k} f_i$. Let, $\text{rank}_{\mathbb{F}(\mathbf{x})}[\mathcal{J}_{\mathbf{x}}(\mathbf{f})] = r$

Claim:
Any $r \times r$ minor of the $\mathcal{J}_{\mathbf{x}}(\mathbf{f})$ can be expressed as a **product of sparse polynomials** with sparsity bounded by $s^{O(k^2)}$.

Proof: Focus on a minor of $\mathcal{J}_{\mathbf{x}}(\mathbf{f})$, let's say

$$\det \begin{pmatrix} \partial_{x_1} f_1 & \dots & \partial_{x_r} f_1 \\ \partial_{x_1} f_2 & \dots & \partial_{x_r} f_2 \\ \vdots & & \\ \partial_{x_1} f_{2k} & \dots & \partial_{x_r} f_{2k} \end{pmatrix}$$

$x_1, \dots, x_r$ occur in at most $2kr$ many P_{ij}'s. So, most of the P_{ij}'s come out common from the determinant.

Finishing off the argument...

- Suppose, $\text{rank}[\mathcal{J}_{\mathbf{x}}(\mathbf{f})] = r \leq m \leq 2k$.
- Then, there's an $r \times r$ non-zero minor of $\mathcal{J}_{\mathbf{x}}(\mathbf{f})$.
- By the previous claim, this minor is a product of sparse polynomials.
- Construct a Ψ (using sparse polynomial hitting set) that preserves nonzeroness of this minor. $\Psi : x_i \mapsto z^{a_i}$.
- This ensures that

$$\text{rank}_{\mathbb{F}(\mathbf{x})} [\mathcal{J}_{\mathbf{x}}(\mathbf{f})] = \text{rank}_{\mathbb{F}(\mathbf{z})} [\Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))].$$

- The final map $\Phi : x_i \mapsto \sum_{j=1}^{2k} y_j t^{ij} + z^{a_i}$ has only $2k+2$ variables. (finally, use sparse polynomial hitting set again)

Finishing off the argument...

- Suppose, $\text{rank}[\mathcal{J}_{\mathbf{x}}(\mathbf{f})] = r \leq m \leq 2k$.
- Then, there's an $r \times r$ non-zero minor of $\mathcal{J}_{\mathbf{x}}(\mathbf{f})$.
- By the previous claim, this minor is a product of sparse polynomials.
- Construct a Ψ (using sparse polynomial hitting set) that preserves nonzeroness of this minor. $\Psi : x_i \mapsto z^{a_i}$.
- This ensures that

$$\text{rank}_{\mathbb{F}(\mathbf{x})}\left[\mathcal{J}_{\mathbf{x}}(\mathbf{f})\right] = \text{rank}_{\mathbb{F}(\mathbf{z})}\left[\Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))\right].$$

- The final map $\Phi : x_i \mapsto \sum_{j=1}^{2k} y_j t^{ij} + z^{a_i}$ has only $2k + 2$ variables. (finally, use sparse polynomial hitting set again)

Finishing off the argument...

- Suppose, $\text{rank}[\mathcal{J}_{\mathbf{x}}(\mathbf{f})] = r \leq m \leq 2k$.
- Then, there's an $r \times r$ non-zero minor of $\mathcal{J}_{\mathbf{x}}(\mathbf{f})$.
- By the previous claim, this minor is a product of sparse polynomials.
- Construct a Ψ (using sparse polynomial hitting set) that preserves nonzeroness of this minor. $\Psi : x_i \mapsto z^{a_i}$.
- This ensures that

$$\text{rank}_{\mathbb{F}(\mathbf{x})}\left[\mathcal{J}_{\mathbf{x}}(\mathbf{f})\right] = \text{rank}_{\mathbb{F}(\mathbf{z})}\left[\Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))\right].$$

- The final map $\Phi : x_i \mapsto \sum_{j=1}^{2k} y_j t^{ij} + z^{a_i}$ has only $2k+2$ variables. (finally, use sparse polynomial hitting set again)

Finishing off the argument...

- Suppose, $\text{rank}[\mathcal{J}_{\mathbf{x}}(\mathbf{f})] = r \leq m \leq 2k$.
- Then, there's an $r \times r$ non-zero minor of $\mathcal{J}_{\mathbf{x}}(\mathbf{f})$.
- By the previous claim, this minor is a product of sparse polynomials.
- Construct a Ψ (using sparse polynomial hitting set) that preserves nonzeroness of this minor. $\Psi : x_i \mapsto z^{a_i}$.
- This ensures that

$$\text{rank}_{\mathbb{F}(\mathbf{x})} \left[\mathcal{J}_{\mathbf{x}}(\mathbf{f})\right] = \text{rank}_{\mathbb{F}(\mathbf{z})} \left[\Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))\right].$$

- The final map $\Phi : x_i \mapsto \sum_{j=1}^{2k} y_j t^{ij} + z^{a_i}$ has only $2k+2$ variables. (finally, use sparse polynomial hitting set again)

Finishing off the argument...

- Suppose, $\mathrm{rank}[\mathcal{J}_{\mathbf{x}}(\mathbf{f})] = r \leq m \leq 2k$.
- Then, there's an $r \times r$ non-zero minor of $\mathcal{J}_{\mathbf{x}}(\mathbf{f})$.
- By the previous claim, this minor is a product of sparse polynomials.
- Construct a Ψ (using sparse polynomial hitting set) that preserves nonzeroness of this minor. $\Psi : x_i \mapsto z^{a_i}$.
- This ensures that

$$\mathrm{rank}_{\mathbb{F}(\mathbf{x})}\,[\mathcal{J}_{\mathbf{x}}(\mathbf{f})] = \mathrm{rank}_{\mathbb{F}(\mathbf{z})}\,[\Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))].$$

- The final map $\Phi : x_i \mapsto \sum_{j=1}^{2k} y_j t^{ij} + z^{a_i}$ has only $2k+2$ variables. (finally, use sparse polynomial hitting set again)

Finishing off the argument...

- Suppose, $\text{rank}[\mathcal{J}_{\mathbf{x}}(\mathbf{f})] = r \leq m \leq 2k$.
- Then, there's an $r \times r$ non-zero minor of $\mathcal{J}_{\mathbf{x}}(\mathbf{f})$.
- By the previous claim, this minor is a product of sparse polynomials.
- Construct a Ψ (using sparse polynomial hitting set) that preserves nonzeroness of this minor. $\Psi : x_i \mapsto z^{a_i}$.
- This ensures that

$$\text{rank}_{\mathbb{F}(\mathbf{x})}\,[\mathcal{J}_{\mathbf{x}}(\mathbf{f})] = \text{rank}_{\mathbb{F}(\mathbf{z})}\,[\Psi(\mathcal{J}_{\mathbf{x}}(\mathbf{f}))].$$

- The final map $\Phi : x_i \mapsto \sum_{j=1}^{2k} y_j t^{ij} + z^{a_i}$ has only $2k+2$ variables. (finally, use sparse polynomial hitting set again)

Depth-4 occur-k lower bound

Recall the theorem...

Any depth-4 occur-k formula that computes Det_n must have size $s = 2^{\Omega(n/k^2)}$.

A certain simplification:

Suppose $\mathrm{Det}_n(M) = C = \sum_{i=1}^{m} f_i$, where $f_i = \prod_{j=1}^{d} P_{ij}$. We can assume w.l.o.g that $m \leq 2k$.

Proof.

Notice that $C(x_1 + 1) - C(x_1)$ is a minor of M and hence a Det_{n-1} polynomial. Argue on this minor. □

Depth-4 occur-k lower bound

Recall the theorem...

Any depth-4 occur-k formula that computes Det_n must have size $s = 2^{\Omega(n/k^2)}$.

A certain simplification:

Suppose $\mathrm{Det}_n(M) = C = \sum_{i=1}^{m} f_i$, where $f_i = \prod_{j=1}^{d} P_{ij}$. We can assume w.l.o.g that $m \leq 2k$.

Proof.

Notice that $C(x_1 + 1) - C(x_1)$ is a minor of M and hence a Det_{n-1} polynomial. Argue on this minor. □

Depth-4 occur-k lower bound

Recall the theorem...

Any depth-4 occur-k formula that computes Det_n must have size $s = 2^{\Omega(n/k^2)}$.

A certain simplification:

Suppose $\mathrm{Det}_n(M) = C = \sum_{i=1}^{m} f_i$, where $f_i = \prod_{j=1}^{d} P_{ij}$. We can assume w.l.o.g that $m \leq 2k$.

Proof.

Notice that $C(x_1 + 1) - C(x_1)$ is a minor of M and hence a Det_{n-1} polynomial. Argue on this minor. □

Depth-4 occur-k lower bound

- Suppose $\text{Det}_n = C = \sum_{i=1}^{2k} f_i$, where $f_i = \prod_{j=1}^{d} P_{ij}$.
- Which means, $\{\text{Det}_n, f_1, \ldots, f_{2k}\}$ are algebraically dependent.
- Hence, $\mathcal{J}_{\mathbf{x}}(\text{Det}_n, f_1, \ldots, f_{2k})$ has rank $< 2k + 1$.
- This means, every $(2k+1) \times (2k+1)$ minor of $\mathcal{J}_{\mathbf{x}}(\text{Det}_n, f_1, \ldots, f_{2k})$ is zero.

Depth-4 occur-k lower bound

- Suppose $\mathrm{Det}_n = C = \sum_{i=1}^{2k} f_i$, where $f_i = \prod_{j=1}^{d} P_{ij}$.
- Which means, $\{\mathrm{Det}_n, f_1, \ldots, f_{2k}\}$ are algebraically dependent.
- Hence, $\mathcal{J}_{\mathbf{x}}(\mathrm{Det}_n, f_1, \ldots, f_{2k})$ has rank $< 2k+1$.
- This means, every $(2k+1) \times (2k+1)$ minor of $\mathcal{J}_{\mathbf{x}}(\mathrm{Det}_n, f_1, \ldots, f_{2k})$ is zero.

Depth-4 occur-k lower bound

- Suppose $\mathrm{Det}_n = C = \sum_{i=1}^{2k} f_i$, where $f_i = \prod_{j=1}^{d} P_{ij}$.
- Which means, $\{\mathrm{Det}_n, f_1, \ldots, f_{2k}\}$ are algebraically dependent.
- Hence, $\mathcal{J}_{\mathbf{x}}(\mathrm{Det}_n, f_1, \ldots, f_{2k})$ has rank $< 2k + 1$.
- This means, every $(2k+1) \times (2k+1)$ minor of $\mathcal{J}_{\mathbf{x}}(\mathrm{Det}_n, f_1, \ldots, f_{2k})$ is zero.

Depth-4 occur-k lower bound

- Suppose $\text{Det}_n = C = \sum_{i=1}^{2k} f_i$, where $f_i = \prod_{j=1}^{d} P_{ij}$.
- Which means, $\{\text{Det}_n, f_1, \ldots, f_{2k}\}$ are algebraically dependent.
- Hence, $\mathcal{J}_{\mathbf{x}}(\text{Det}_n, f_1, \ldots, f_{2k})$ has rank $< 2k + 1$.
- This means, every $(2k+1) \times (2k+1)$ minor of $\mathcal{J}_{\mathbf{x}}(\text{Det}_n, f_1, \ldots, f_{2k})$ is zero.

Depth-4 occur-k lower bound

- Fix one such minor.

$$\det \begin{pmatrix} \partial_{x_1} Det_n & \cdots & \partial_{x_{2k+1}} Det_n \\ \partial_{x_1} f_1 & \cdots & \partial_{x_{2k+1}} f_1 \\ \vdots & \ddots & \vdots \\ \partial_{x_1} f_{2k} & \cdots & \partial_{x_{2k+1}} f_{2k} \end{pmatrix} = 0$$

$$\Rightarrow \sum_{\ell=1}^{2k+1} g_\ell \cdot M_\ell = 0,$$

where g_ℓ's are sparse polynomials and M_ℓ's are **principal minors** of M. (g_ℓ's are sparse as most of the P_{ij}'s come out common from the above determinant.)

Depth-4 occur-k lower bound

- Fix one such minor.

$$\det \begin{pmatrix} \partial_{x_1} Det_n & \cdots & \partial_{x_{2k+1}} Det_n \\ \partial_{x_1} f_1 & \cdots & \partial_{x_{2k+1}} f_1 \\ \vdots & \ddots & \vdots \\ \partial_{x_1} f_{2k} & \cdots & \partial_{x_{2k+1}} f_{2k} \end{pmatrix} = 0$$

$$\Rightarrow \sum_{\ell=1}^{2k+1} g_\ell \cdot M_\ell = 0,$$

where g_ℓ's are sparse polynomials and M_ℓ's are **principal minors** of M. (g_ℓ's are sparse as most of the P_{ij}'s come out common from the above determinant.)

Depth-4 occur-k lower bound

Can such 'sparse-minor identities' exist?

Theorem:
If $\sum_{\ell=1}^{t} g_\ell \cdot M_\ell = 0$ then the total sparsity of the g_ℓ's is $\Omega(2^{n/2-t})$.

Proof.
The $t = 2$ case:
If $g_1 M_1 = -g_2 M_2$ then $M_1 | g_2$, as M_1 is irreducible (so is M_2).
The general t case involves a more careful combinatorial argument.
We'll skip it here. □

Note: The above theorem is tight in the sense that such identities do exist for $t = n$.

Depth-4 occur-k lower bound

Can such 'sparse-minor identities' exist?

Theorem:
If $\sum_{\ell=1}^{t} g_\ell \cdot M_\ell = 0$ then the total sparsity of the g_ℓ's is $\Omega(2^{n/2-t})$.

Proof.
The $t = 2$ case:
If $g_1 M_1 = -g_2 M_2$ then $M_1 | g_2$, as M_1 is irreducible (so is M_2).
The general t case involves a more careful combinatorial argument.
We'll skip it here. □

Note: The above theorem is tight in the sense that such identities do exist for $t = n$.

Depth-4 occur-k lower bound

Can such 'sparse-minor identities' exist?

Theorem:
If $\sum_{\ell=1}^{t} g_\ell \cdot M_\ell = 0$ then the total sparsity of the g_ℓ's is $\Omega(2^{n/2-t})$.

Proof.
The $t = 2$ case:
If $g_1 M_1 = -g_2 M_2$ then $M_1 | g_2$, as M_1 is irreducible (so is M_2).
The general t case involves a more careful combinatorial argument.
We'll skip it here. □

Note: The above theorem is tight in the sense that such identities do exist for $t = n$.

Depth-4 occur-k lower bound

Can such 'sparse-minor identities' exist?

Theorem:
If $\sum_{\ell=1}^{t} g_\ell \cdot M_\ell = 0$ then the total sparsity of the g_ℓ's is $\Omega(2^{n/2-t})$.

Proof.
The $t = 2$ case:
If $g_1 M_1 = -g_2 M_2$ then $M_1 | g_2$, as M_1 is irreducible (so is M_2).
The general t case involves a more careful combinatorial argument.
We'll skip it here. □

Note: The above theorem is tight in the sense that such identities do exist for $t = n$.

Depth-4 occur-k lower bound

Can such 'sparse-minor identities' exist?

Theorem:
If $\sum_{\ell=1}^{t} g_\ell \cdot M_\ell = 0$ then the total sparsity of the g_ℓ's is $\Omega(2^{n/2-t})$.

Proof.
The $t = 2$ case:
If $g_1 M_1 = -g_2 M_2$ then $M_1 | g_2$, as M_1 is irreducible (so is M_2).
The general t case involves a more careful combinatorial argument.
We'll skip it here. □

Note: The above theorem is tight in the sense that such identities do exist for $t = n$.

Thank you!