Multiple Facets of Algebraic Computation

C Ramya
IMSc, Chennai.

September 2023
Algebraic Computation

- Goal: Understand the amount of *computational resource* required to solve a given *computational problem* on a *computational model*. 

Objects: Polynomials. E.g., $f = x^2 + 3x + x^2 - 2$. 

Resource: No. of arithmetic operations (+, ×) 

Model: Arithmetic circuits: DAGs with leaves labelled by variables or constants (from $F$) and internal gates labelled by {+, ×}. 

Algebraic Computation

- **Goal:** Understand the amount of *computational resource* required to solve a given *computational problem* on a *computational model*.
- **Objects:** Polynomials. E.g., $f = x_1^2 + 3x_1x_2 - 2$. 
Algebraic Computation

- **Goal:** Understand the amount of *computational resource* required to solve a given *computational problem* on a *computational model*.

- **Objects:** Polynomials. E.g., \( f = x_1^2 + 3x_1x_2 - 2 \).

- **Resource:** No. of arithmetic operations (+, \( \times \)).
Algebraic Computation

- **Goal:** Understand the amount of *computational resource* required to solve a given *computational problem* on a *computational model*.

- **Objects:** Polynomials. E.g., \( f = x_1^2 + 3x_1x_2 - 2 \).
  
  **Resource:** No. of arithmetic operations (+, ×)

- **Model:** Arithmetic circuits: DAGs with leaves labelled by variables or constants (from \( \mathbb{F} \)) and internal gates labelled by \{ +, × \}.
Complexity of polynomials

- $\text{size}(C)$ - number of gates in circuit $C \equiv$ no. of arithmetic operations to compute $f$. 

$\sum_{n=1}^{\infty} x_1 + x_2 + \cdots + x_n$ 

$\sum_1^{\infty} = x_1$ 

$\sum_2^{\infty} = x_2$ 

$\sum_3^{\infty} = x_3$ 

$\sum_n^{\infty} = x_n$ 

$\sum_{\infty} = \sum_1^{\infty} + \sum_2^{\infty} + \sum_3^{\infty} + \sum_n^{\infty}$ 

$\text{size}(C_1) = 1$ 

$\text{size}(C_2) = 1$ 

$\text{size}(C_3) = 2$ 

$\text{size}(C_n) = n - 1$ 

$C_1$ 

$C_2$ 

$C_3$ 

$C_n$ 

$\sum_{n=1}^{\infty}$ is a polynomial family.
Complexity of polynomials

- \( \text{size}(C) \) - number of gates in circuit \( C \) \( \equiv \) no. of arithmetic operations to compute \( f \).
- \( \text{depth}(C) \) - length of longest path from input to output gate of \( C \).

There can be several different circuits computing a given polynomial.

\[ \text{SUM}_n = x_1 + x_2 + \cdots + x_n. \]
Complexity of polynomials

- \textit{size}(C) - number of gates in circuit \( C \) \equiv no. of arithmetic operations to compute \( f \).
- \textit{depth}(C) - length of longest path from input to output gate of \( C \).
- There can be several different circuits computing a given polynomial.
Complexity of polynomials

- $size(C)$ - number of gates in circuit $C \equiv$ no. of arithmetic operations to compute $f$.
- $depth(C)$ - length of longest path from input to output gate of $C$.
- There can be several different circuits computing a given polynomial.
Complexity of polynomials

- \(\text{size}(C)\) - number of gates in circuit \(C\) \(\equiv\) no. of arithmetic operations to compute \(f\).
- \(\text{depth}(C)\) - length of longest path from input to output gate of \(C\).

There can be several different circuits computing a given polynomial. Let \(S\sum_{n} = x_1 + x_2 + \cdots + x_n\).
Complexity of polynomials

- $\text{size}(C)$ - number of gates in circuit $C \equiv$ no. of arithmetic operations to compute $f$.
- $\text{depth}(C)$ - length of longest path from input to output gate of $C$.
- There can be several different circuits computing a given polynomial.

Let $\text{SUM}_n = x_1 + x_2 + \cdots + x_n$.

\[
\begin{align*}
  \text{SUM}_1 & = x_1 + 0 \\
  \text{SUM}_2 & = x_1 + x_2 \\
  \text{SUM}_3 & = x_1 + x_2 + x_3 \\
  \vdots & \\
  \text{SUM}_n & = x_1 + x_2 + \cdots + x_n
\end{align*}
\]

- $C_1$, $\text{size}(C_1) = 1$
- $C_2$, $\text{size}(C_2) = 1$
- $C_3$, $\text{size}(C_3) = 2$
- $C_n$, $\text{size}(C_n) = n - 1$

$\text{SUM} = (\text{SUM}_n)_{n \geq 1}$ is a polynomial family.
Valiant’s Conjecture

- Any $n$-variate degree-$d$ polynomial can be computed by...
Valiant’s Conjecture

- Any $n$-variate degree-$d$ polynomial can be computed by a depth two circuit of size $O\left(\binom{n+d}{d}\right)$.

There exists $n$-variate degree-$d$ polynomials that require arithmetic circuits of size $\Omega(n^d + d^d)$.

A polynomial family $(f_n)_{n \geq 1}$ is efficiently computable if for every $n$, $\deg(f_n) = \text{poly}(n)$ and there is poly($n$) size circuit for $f_n$.

E.g., SUM, Symbolic Determinant $\text{det} = (\text{det}_n)_{n \geq 1}$.

Class VP: class of efficiently computable polynomial families.

Are there polynomials that are hard to compute (outside VP)? YES.

Goal: Find an explicit polynomial outside VP.

Explicit: coefficient of any monomial is reasonably easy to compute.

Valiant’s Conjecture: Any circuit for $\text{perm}_n$ requires size $n^{\omega(1)}$. 
Valiant’s Conjecture

- Any $n$-variate degree-$d$ polynomial can be computed by a depth two circuit of size $O\left(\binom{n+d}{d}\right)$.
- There exists $n$-variate degree-$d$ polynomials that require arithmetic circuits of size $\Omega\left(\sqrt{\binom{n+d}{d}}\right)$. 
Valiant’s Conjecture

- Any $n$-variate degree-$d$ polynomial can be computed by a depth two circuit of size $O\left(\binom{n+d}{d}\right)$.
- There exists $n$-variate degree-$d$ polynomials that require arithmetic circuits of size $\Omega\left(\sqrt{\binom{n+d}{d}}\right)$.
- A polynomial family $(f_n)_{n \geq 1}$ is efficiently computable if for every $n$, $\deg(f_n) = \text{poly}(n)$ and there is poly$(n)$ size circuit for $f_n$. 

E.g.: $\text{SUM}$, Symbolic Determinant $\text{det} = \text{det}_n$ for $n \geq 1$.

Class $\text{VP}$: class of efficiently computable polynomial families.

Are there polynomials that are hard to compute (outside $\text{VP}$)? YES.

Goal: Find an explicit polynomial outside $\text{VP}$. 

Explicit: coefficient of any monomial is reasonably easy to compute.
Valiant’s Conjecture

- Any $n$-variate degree-$d$ polynomial can be computed by a depth two circuit of size $O\left(\binom{n+d}{d}\right)$.
- There exists $n$-variate degree-$d$ polynomials that require arithmetic circuits of size $\Omega\left(\sqrt{\binom{n+d}{d}}\right)$.
- A polynomial family $(f_n)_{n \geq 1}$ is efficiently computable if for every $n$, $\deg(f_n) = \text{poly}(n)$ and there is poly$(n)$ size circuit for $f_n$. E.g.,: SUM, Symbolic Determinant $\det = (\det_n)_{n \geq 1}$.  

Class VP: class of efficiently computable polynomial families.
Valiant’s Conjecture

- Any $n$-variate degree-$d$ polynomial can be computed by a depth two circuit of size $O\left(\binom{n+d}{d}\right)$.
- There exists $n$-variate degree-$d$ polynomials that require arithmetic circuits of size $\Omega\left(\sqrt[n]{\binom{n+d}{d}}\right)$.
- A polynomial family $(f_n)_{n \geq 1}$ is efficiently computable if for every $n$, $\deg(f_n) = \text{poly}(n)$ and there is poly$(n)$ size circuit for $f_n$. E.g.,: SUM, Symbolic Determinant $\det = (\det_n)_{n \geq 1}$.
- **Class** VP: class of efficiently computable polynomial families.
Valiant’s Conjecture

- Any $n$-variate degree-$d$ polynomial can be computed by a depth two circuit of size $O\left(\binom{n+d}{d}\right)$.

- There exists $n$-variate degree-$d$ polynomials that require arithmetic circuits of size $\Omega\left(\sqrt{\binom{n+d}{d}}\right)$.

- A polynomial family $(f_n)_{n \geq 1}$ is efficiently computable if for every $n$, $\deg(f_n) = \text{poly}(n)$ and there is poly($n$) size circuit for $f_n$. E.g.,: SUM, Symbolic Determinant $\det = (\det_n)_{n \geq 1}$.

- **Class** VP: class of efficiently computable polynomial families.

- Are there polynomials that are hard to compute (outside VP)? YES.
Valiant’s Conjecture

- Any \( n \)-variate degree-\( d \) polynomial can be computed by a depth two circuit of size \( O\left(\binom{n+d}{d}\right)\).
- There exists \( n \)-variate degree-\( d \) polynomials that require arithmetic circuits of size \( \Omega\left(\sqrt{\binom{n+d}{d}}\right)\).
- A polynomial family \( (f_n)_{n \geq 1} \) is efficiently computable if for every \( n \), \( \text{deg}(f_n) = \text{poly}(n) \) and there is \( \text{poly}(n) \) size circuit for \( f_n \).
  E.g.,: SUM, Symbolic Determinant \( \det = (\det_n)_{n \geq 1} \).
- **Class** VP: class of efficiently computable polynomial families.
- Are there polynomials that are hard to compute(outside VP)? YES.
- **Goal**: Find an explicit polynomial outside VP.
  Explicit: coefficient of any monomial is reasonably easy to compute.
Valiant’s Conjecture

- Any \( n \)-variate degree-\( d \) polynomial can be computed by a depth two circuit of size \( O \left( \binom{n+d}{d} \right) \).
- There exists \( n \)-variate degree-\( d \) polynomials that require arithmetic circuits of size \( \Omega \left( \sqrt{\binom{n+d}{d}} \right) \).
- A polynomial family \( (f_n)_{n \geq 1} \) is \emph{efficiently computable} if for every \( n \), \( \text{deg}(f_n) = \text{poly}(n) \) and there is \( \text{poly}(n) \) size circuit for \( f_n \). E.g.,: SUM, Symbolic Determinant \( \det = (\det_n)_{n \geq 1} \).
- \textbf{Class VP}: class of efficiently computable polynomial families.
- Are there polynomials that are \emph{hard} to compute(outside VP)? YES.
- \textbf{Goal}: Find an \emph{explicit polynomial} outside VP. Explicit: coefficient of any monomial is \emph{reasonably} easy to compute.

\textbf{Valiant’s Conjecture}: Any circuit for \( \text{perm}_n \) requires size \( n^{\omega(1)} \).
Towards Valiant’s Hypothesis

(Baur, Strassen ‘83) Any circuit for $x_1^d + \cdots + x_n^d$ requires size $\Omega(n \log d)$. 
Towards Valiant’s Hypothesis

(Baur, Strassen ‘83) Any circuit for $x_1^d + \cdots + x_n^d$ requires size $\Omega(n \log d)$.
(Folklore) Any depth-2 circuit computing perm$_n$ requires size $n!$. 
Towards Valiant’s Hypothesis

(Baur,Strassen’83) Any circuit for $x_1^d + \cdots + x_n^d$ requires size $\Omega(n \log d)$.

(Folklore) Any depth-2 circuit computing $\text{perm}_n$ requires size $n!$.

Depth reduction $n^{\omega(\sqrt{d})}$ lower bound for depth-three circuits computing an explicit $n$-variate, degree $d$ polynomial is sufficient to resolve Valiant’s conjecture.
Towards Valiant’s Hypothesis

(Baur, Strassen ‘83) Any circuit for $x_1^d + \cdots + x_n^d$ requires size $\Omega(n \log d)$.

(Folklore) Any depth-2 circuit computing $\text{perm}_n$ requires size $n!$.

Depth reduction $n^{\omega(\sqrt{d})}$ lower bound for depth-three circuits computing an explicit $n$-variate, degree $d$ polynomial is sufficient to resolve Valiant’s conjecture.

(Limaye, Srinivasan, Tavenas ’21) There is an explicit $n$-variate polynomial (in VP) of degree $d$ such that any depth three circuit for it has size $n^{\Omega(\sqrt{d})}$. 
Towards Valiant’s Hypothesis

(Baur, Strassen ‘83) Any circuit for \( x_1^d + \cdots + x_n^d \) requires size \( \Omega(n \log d) \).

(Folklore) Any depth-2 circuit computing \( \text{perm}_n \) requires size \( n! \).

Depth reduction \( n^{\omega(\sqrt{d})} \) lower bound for depth-three circuits computing an explicit \( n \)-variate, degree \( d \) polynomial is sufficient to resolve Valiant’s conjecture.

(Limaye, Srinivasan, Tavenas ’21) There is an explicit \( n \)-variate polynomial (in VP) of degree \( d \) such that any depth three circuit for it has size \( n^{\Omega(\sqrt{d})} \).

Perhaps the principal embarrassment of complexity theory at the present time is its failure to provide techniques for proving non-trivial lower bounds on the complexity of some of the commonest combinatorial and arithmetic problems.
Towards Valiant’s Hypothesis

(Baur, Strassen ‘83) Any circuit for $x_1^d + \cdots + x_n^d$ requires size $\Omega(n \log d)$.  
(Folklore) Any depth-2 circuit computing $\text{perm}_n$ requires size $n!$.  

Depth reduction $n^{\omega(\sqrt{d})}$ lower bound for depth-three circuits computing an explicit $n$-variate, degree $d$ polynomial is sufficient to resolve Valiant’s conjecture.

(Limaye, Srinivasan, Tavenas ’21) There is an explicit $n$-variate polynomial (in VP) of degree $d$ such that any depth three circuit for it has size $n^{\Omega(\sqrt{d})}$.

Perhaps the principal embarrassment of complexity theory at the present time is its failure to provide techniques for proving non-trivial lower bounds on the complexity of some of the commonest combinatorial and arithmetic problems.

- Valiant (1975)
Proving Lower Bounds: A Toy Example

Let $\mathcal{C} = \{(\alpha x - \beta)^2 \mid \alpha, \beta \in \mathbb{C}\}$. **Goal:** Find an explicit $h(x) \not\in \mathcal{C}$. 

Let $f(x) = ax^2 + bx + c$ be any quadratic polynomial. Then, $f(x) \in \mathcal{C}$ if and only if $b^2 - 4ac = 0$. Note, $\text{coeff}(f) = (a, b, c)$.

Find a polynomial $h(x) = ax^2 + bx + c$ with non-zero discriminant.

Consider $P(z_1, z_2, z_3) = z_2^2 - 4z_1z_3$. Then, $f(x) \in \mathcal{C} \Rightarrow P(\text{coeff}(f)) = 0$.

$P(z_1, z_2, z_3)$ is efficiently computable. There is polynomial $h(x) \in \mathbb{F}[x]$ such that $P(\text{coeff}(h)) \neq 0$.

Proving Lower Bounds against $\mathcal{C}$: Find a property $P$ that every polynomial in $\mathcal{C}$ satisfies and then find an explicit $h(x) \not\in \mathcal{C}$. 

Goal: $h(\bar{x}) \not\in \text{VP}$
Proving Lower Bounds: A Toy Example

Let $C = \{ (\alpha x - \beta)^2 \mid \alpha, \beta \in \mathbb{C} \}$. **Goal:** Find an explicit $h(x) \not\in C$.

- Let $f(x) = ax^2 + bx + c$ be any quadratic polynomial. Then, $f(x)$ has a repeated root if and only if $b^2 - 4ac = 0$. Note, $\text{coeff}(f) = (a, b, c)$. 
Proving Lower Bounds: A Toy Example

Let $\mathcal{C} = \{(\alpha x - \beta)^2 \mid \alpha, \beta \in \mathbb{C}\}$. **Goal:** Find an explicit $h(x) \not\in \mathcal{C}$.

- Let $f(x) = ax^2 + bx + c$ be any quadratic polynomial. Then, $f(x)$ has a repeated root if and only if $b^2 - 4ac = 0$. Note, $\text{coeff}(f) = (a, b, c)$.
- Find a polynomial $h(x) = ax^2 + bx + c$ with non-zero discriminant.
Proving Lower Bounds: A Toy Example

Let $C = \{(\alpha x - \beta)^2 \mid \alpha, \beta \in \mathbb{C}\}$. **Goal:** Find an explicit $h(x) \notin C$.

- Let $f(x) = ax^2 + bx + c$ be any quadratic polynomial. Then, $f(x)$ has a repeated root if and only if $b^2 - 4ac = 0$. Note, $\text{coeff}(f) = (a, b, c)$.
- Find a polynomial $h(x) = ax^2 + bx + c$ with non-zero discriminant.

Consider $P(z_1, z_2, z_3) = z_2^2 - 4z_1z_3$. Then,

- $f(x) \in C \Rightarrow P(\text{coeff}(f)) = 0$.
- $P(z_1, z_2, z_3)$ is efficiently computable.
- There is polynomial $h(x) \in \mathbb{F}[x]^{\leq 2}$ such that $P(\text{coeff}(h)) \neq 0$. 
Proving Lower Bounds: A Toy Example

Let $\mathcal{C} = \{(\alpha x - \beta)^2 | \alpha, \beta \in \mathbb{C}\}$. **Goal:** Find an explicit $h(x) \not\in \mathcal{C}$.

- Let $f(x) = ax^2 + bx + c$ be any quadratic polynomial. Then, $f(x)$ has a repeated root if and only if $b^2 - 4ac = 0$. Note, $\text{coeff}(f) = (a, b, c)$.
- Find a polynomial $h(x) = ax^2 + bx + c$ with non-zero discriminant.

Consider $P(z_1, z_2, z_3) = z_2^2 - 4z_1z_3$. Then,

- $f(x) \in \mathcal{C} \implies P(\text{coeff}(f)) = 0$.
- $P(z_1, z_2, z_3)$ is efficiently computable.
- There is polynomial $h(x) \in \mathbb{F}[x]^{\leq 2}$ such that $P(\text{coeff}(h)) \neq 0$.

**Proving Lower Bounds against $\mathcal{C}$:** Find a property $P$ that every polynomial in $\mathcal{C}$ satisfies and then find an explicit $h$ that does not satisfy $P$. 

Goal: $h(\bar{x}) \not\in \text{VP}$

$h(\bar{x})$ explicit
Another Toy Example

Let $C$ be the class of $\Sigma\Pi$-circuits. Any $\Sigma\Pi$ circuit computing the permanent requires size $n!$. 

Define $\mu : \mathbb{R}[x_1, \ldots, x_n] \to \mathbb{R}$ s.t. $\mu(f_1 + \cdots + f_s) \leq \mu(f_1) + \cdots + \mu(f_s)$. E.g., $\mu(f) \equiv$ number of monomials of $f$. 

Observe that $\mu(\text{perm}_n) = n!$. Therefore, $s \geq n!$. For more (in fact most) sophisticated circuit classes $C$:

1. Construct a measure $\mu : \mathbb{R}[x_1, \ldots, x_n] \to \mathbb{R}$: $\mu(f)$ is small for $f \in C$. 
2. $\mu(h)$ is large for an explicit polynomial $h$. 
3. $\mu(f)$ is rank($M_f$) for a matrix $M_f$ associated with polynomial $f$. 

Another Toy Example

Let $C$ be the class of $\Sigma\Pi$-circuits. Any $\Sigma\Pi$ circuit computing the permanent requires size $n!$.

- Let $f$ be computable by a $\Sigma\Pi$ circuit of top fanin $s$.

Define $\mu : F[x_1, \ldots, x_n] \rightarrow \mathbb{R}$ s.t. $\mu(f_1 + \cdots + f_s) \leq \mu(f_1) + \cdots + \mu(f_s)$.

E.g., $\mu(f) \equiv \text{number of monomials of } f$.

Observe that $\mu(\text{perm}_n) = n!$.

Therefore, $s \geq n!$.

For more (in fact most) sophisticated circuit classes $C$:

- Construct a measure $\mu : F[x_1, \ldots, x_n] \rightarrow \mathbb{R}$: $\mu(f)$ is small for $f \in C$.
- $\mu(h)$ is large for an explicit polynomial $h$.
- $\mu(f)$ is rank($M_f$) for a matrix $M_f$ associated with polynomial $f$.
Another Toy Example

Let $C$ be the class of $\Sigma \Pi$-circuits. Any $\Sigma \Pi$ circuit computing the permanent requires size $n!$.

- Let $f$ be computable by a $\Sigma \Pi$ circuit of top fanin $s$.
- Define $\mu : \mathbb{F}[x_1, \ldots, x_n] \rightarrow \mathbb{R}$ s.t. $\mu(f_1 + \cdots + f_s) \leq \mu(f_1) + \cdots + \mu(f_s)$. 

E.g., $\mu(f) \equiv$ number of monomials of $f$.
Another Toy Example

Let $C$ be the class of $\Sigma\Pi$-circuits. Any $\Sigma\Pi$ circuit computing the permanent requires size $n!$.

- Let $f$ be computable by a $\Sigma\Pi$ circuit of top fanin $s$.
- Define $\mu : \mathbb{F}[x_1, \ldots, x_n] \to \mathbb{R}$ s.t. $\mu(f_1 + \cdots + f_s) \leq \mu(f_1) + \cdots + \mu(f_s)$.
  E.g., $\mu(f) \triangleq$ number of monomials of $f$. 

Observe that $\mu(\text{perm}_n) = n!$.

Therefore, $s \geq n!$.

For more (in fact most) sophisticated circuit classes $C$:
- Construct a measure $\mu : \mathbb{F}[x_1, \ldots, x_n] \to \mathbb{R}$: $\mu(f)$ is small for $f \in C$.
- $\mu(h)$ is large for an explicit polynomial $h$.
- $\mu(f) \triangleq \text{rank}(M_f)$ for a matrix $M_f$ associated with polynomial $f$. 

7 / 19
Another Toy Example

Let $C$ be the class of $\Sigma\Pi$-circuits. Any $\Sigma\Pi$ circuit computing the permanent requires size $n!$.

- Let $f$ be computable by a $\Sigma\Pi$ circuit of top fanin $s$.
- Define $\mu : \mathbb{F}[x_1, \ldots, x_n] \rightarrow \mathbb{R}$ s.t. $\mu(f_1 + \cdots + f_s) \leq \mu(f_1) + \cdots + \mu(f_s)$. E.g., $\mu(f) \triangleq$ number of monomials of $f$.

$$
\mu(f) = \mu(m_1 + \cdots + m_s) \\
\leq \mu(m_1) + \cdots + \mu(m_s) \leq s
$$
Another Toy Example

Let $C$ be the class of $\Sigma\Pi$-circuits. Any $\Sigma\Pi$ circuit computing the permanent requires size $n!$.

- Let $f$ be computable by a $\Sigma\Pi$ circuit of top fanin $s$.
- Define $\mu : \mathbb{F}[x_1, \ldots, x_n] \rightarrow \mathbb{R}$ s.t. $\mu(f_1 + \cdots + f_s) \leq \mu(f_1) + \cdots + \mu(f_s)$. E.g., $\mu(f) \triangleq$ number of monomials of $f$.
  \[
  \mu(f) = \mu(m_1 + \cdots + m_s) \\
  \leq \mu(m_1) + \cdots + \mu(m_s) \leq s
  \]
- Observe that $\mu(\text{perm}_n) = n!$
Another Toy Example

Let $C$ be the class of $\Sigma\Pi$-circuits. Any $\Sigma\Pi$ circuit computing the permanent requires size $n!$.

- Let $f$ be computable by a $\Sigma\Pi$ circuit of top fanin $s$.
- Define $\mu : \mathbb{F}[x_1, \ldots, x_n] \rightarrow \mathbb{R}$ s.t. $\mu(f_1 + \cdots + f_s) \leq \mu(f_1) + \cdots + \mu(f_s)$.  
  E.g., $\mu(f) \triangleq$ number of monomials of $f$.

$$
\mu(f) = \mu(m_1 + \cdots + m_s) \\
\leq \mu(m_1) + \cdots + \mu(m_s) \leq s
$$

- Observe that $\mu(perm_n) = n!$ Therefore, $s \geq n!$.
Another Toy Example

Let $C$ be the class of $\Sigma \Pi$-circuits. Any $\Sigma \Pi$ circuit computing the permanent requires size $n!$.

- Let $f$ be computable by a $\Sigma \Pi$ circuit of top fanin $s$.
- Define $\mu : \mathbb{F}[x_1, \ldots, x_n] \to \mathbb{R}$ s.t. $\mu(f_1 + \cdots + f_s) \leq \mu(f_1) + \cdots + \mu(f_s)$.
  E.g., $\mu(f) \triangleq$ number of monomials of $f$.

\[
\mu(f) = \mu(m_1 + \cdots + m_s) \\
\leq \mu(m_1) + \cdots + \mu(m_s) \leq s
\]

- Observe that $\mu(\text{perm}_n) = n!$. Therefore, $s \geq n!$.

For more (in fact most) sophisticated circuit classes $C$:

Construct a measure $\mu : \mathbb{F}[x_1, \ldots, x_n] \to \mathbb{R}$:

- $\mu(f)$ is small for $f \in C$.
- $\mu(h)$ is large for an explicit polynomial $h$. 
Another Toy Example

Let $C$ be the class of $\Sigma \Pi$-circuits. Any $\Sigma \Pi$ circuit computing the permanent requires size $n!$.

- Let $f$ be computable by a $\Sigma \Pi$ circuit of top fanin $s$.
- Define $\mu : \mathbb{F}[x_1, \ldots, x_n] \rightarrow \mathbb{R}$ s.t. $\mu(f_1 + \cdots + f_s) \leq \mu(f_1) + \cdots + \mu(f_s)$. E.g., $\mu(f) \triangleq$ number of monomials of $f$.

\[
\mu(f) = \mu(m_1 + \cdots + m_s) \\
\leq \mu(m_1) + \cdots + \mu(m_s) \leq s
\]

- Observe that $\mu(\text{perm}_n) = n!$ Therefore, $s \geq n!$.

For more (in fact most) sophisticated circuit classes $C$:

Construct a measure $\mu : \mathbb{F}[x_1, \ldots, x_n] \rightarrow \mathbb{R}$:

- $\mu(f)$ is small for $f \in C$.
- $\mu(h)$ is large for an explicit polynomial $h$.
- $\mu(f)$ is $\text{rank}(M_f)$ for a matrix $M_f$ associated with polynomial $f$. 
Proving Lower Bounds against $\mathcal{C}$

Most lower bound proofs against $\mathcal{C}$ construct a measure $\mu : \mathbb{F}[\bar{x}] \rightarrow \mathbb{R}$:

- $\mu(f)$ is small for $f \in \mathcal{C}$ (i.e., $\text{rank}(M_f)$ is small)
- $\mu(h)$ is large for an explicit polynomial $h$. (i.e., $\text{rank}(M_h)$ is large)
Proving Lower Bounds against $C$

Most lower bound proofs against $C$ construct a measure $\mu : \mathbb{F}[\bar{x}] \rightarrow \mathbb{R}$:

- $\mu(f)$ is small for $f \in C$ (i.e., $\text{rank}(M_f)$ is small)
- $\mu(h)$ is large for an explicit polynomial $h$. (i.e., $\text{rank}(M_h)$ is large)

$$M_f = \begin{bmatrix} W_f \end{bmatrix} \quad \text{exists submatrix } W \quad \text{s.t. } \det(W_f) = 0$$

$$M_h = \begin{bmatrix} W_h \end{bmatrix} \quad \det(W_h) \neq 0$$
Most lower bound proofs against $\mathcal{C}$ construct a measure $\mu : \mathbb{F}[\bar{x}] \rightarrow \mathbb{R}$:
- $\mu(f)$ is \textit{small} for $f \in \mathcal{C}$ (i.e., $\text{rank}(M_f)$ is \textit{small})
- $\mu(h)$ is \textit{large} for an explicit polynomial $h$. (i.e., $\text{rank}(M_h)$ is \textit{large})

For $f \in \mathbb{F}[x_1, \ldots, x_n]$ of degree $d = \text{poly}(n)$: $M_f \in \mathbb{F}^{N \times N}$, $N = \binom{n+d}{n}$.
- $M_f[x^\alpha, x^\beta] = \text{coefficient of } x^\alpha \text{ in } \frac{\partial f}{\partial x^\beta}$.
- Entries of $M_f$ are linear in the coefficients of $f$. 

\[
\begin{align*}
M_f &= \begin{bmatrix} W_f \end{bmatrix} \quad \text{s.t. } \text{det}(W_f) = 0 \\
M_h &= \begin{bmatrix} W_h \end{bmatrix} \quad \text{det}(W_h) \neq 0
\end{align*}
\]
Towards Algebraically Natural Proofs

Let \( f \in \mathbb{F}[x_1, \ldots, x_n] \) of degree \( d \). Then, \( f = \sum c_m \cdot m \).

Coefficient-vector: \( \text{coeff}(f) = (c_1, c_2, \ldots, c_N) \in \mathbb{F}^N \) where \( N = \binom{n+d}{n} \).

“Natural” lower bound proof for \( C \subseteq \mathbb{F}[x_1, \ldots, x_n]^{\leq d} \):

\[ C \text{ has a } \textit{natural proof} \text{ if there is a non-zero polynomial } P(z_1, \ldots, z_N): \]

1. **Usefulness**: \( \forall f \in C, P(\text{coeff}(f)) = 0. \)
2. **Constructivity**: \( P \) has degree \( \text{poly}(N) \) and size \( \text{poly}(N) \).
3. **Largeness**: \( P(\text{coeff}(h)) \neq 0 \) for candidate hard polynomial \( h \) (and for many more polynomials).
Towards Algebraically Natural Proofs

Let \( f \in \mathbb{F}[x_1, \ldots, x_n] \) of degree \( d \). Then, \( f = \sum c_m \cdot m \).

Coefficient-vector: \( \text{coeff}(f) = (c_1, c_2, \ldots, c_N) \in \mathbb{F}^N \) where \( N = \binom{n+d}{n} \).

“Natural” lower bound proof for \( C \subseteq \mathbb{F}[x_1, \ldots, x_n]^{\leq d} \):

\( C \) has a natural proof if there is a non-zero polynomial \( P(z_1, \ldots, z_N) \):

1. **Usefulness**: \( \forall f \in C, P(\text{coeff}(f)) = 0 \).
2. **Constructivity**: \( P \) has degree \( \text{poly}(N) \) and size \( \text{poly}(N) \).
3. **Largeness**: \( P(\text{coeff}(h)) \neq 0 \) for candidate hard polynomial \( h \) (and for many more polynomials).

**Example 1**: \( C = \{(\alpha x - \beta)^2 \mid \alpha, \beta \in \mathbb{C} \} \). Then, \( P(z_1, z_2, z_3) = z_2^2 - 4z_1 z_3 \) such that \( P(\text{coeff}(f)) = 0 \) for all \( f \in C \).
Towards Algebraically Natural Proofs

Let $f \in \mathbb{F}[x_1, \ldots, x_n]$ of degree $d$. Then, $f = \sum c_m \cdot m$.

Coefficient-vector: $\text{coeff}(f) = (c_1, c_2, \ldots, c_N) \in \mathbb{F}^N$ where $N = \binom{n+d}{n}$.

“Natural” lower bound proof for $C \subseteq \mathbb{F}[x_1, \ldots, x_n]^{\leq d}$:

$C$ has a natural proof if there is a non-zero polynomial $P(z_1, \ldots, z_N)$:

1. **Usefulness**: $\forall f \in C$, $P(\text{coeff}(f)) = 0$.
2. **Constructivity**: $P$ has degree $\text{poly}(N)$ and size $\text{poly}(N)$.
3. **Largeness**: $P(\text{coeff}(h)) \neq 0$ for candidate hard polynomial $h$ (and for many more polynomials).

**Example 1**: $C = \{(\alpha x - \beta)^2 \mid \alpha, \beta \in \mathbb{C}\}$. Then, $P(z_1, z_2, z_3) = z_2^2 - 4z_1z_3$ such that $P(\text{coeff}(f)) = 0$ for all $f \in C$.

**Example 2**: $C = \{\Sigma \Pi \Sigma, \Sigma \Pi \Sigma \Pi, \Sigma \land \Sigma\}$. Then, $P(z_1, \ldots, z_N) = \det(W)$ such that $P(\text{coeff}(f)) = 0$ for all $f \in C$. 
Towards Algebraically Natural Proofs

Let \( f \in \mathbb{F}[x_1, \ldots, x_n] \) of degree \( d \). Then, \( f = \sum c_m \cdot m \).

Coefficient-vector: \( \text{coeff}(f) = (c_1, c_2, \ldots, c_N) \in \mathbb{F}^N \) where \( N = \binom{n+d}{n} \).

“Natural” lower bound proof for \( C \subseteq \mathbb{F}[x_1, \ldots, x_n]^{\leq d} \):

\[ C \text{ has a natural proof if there is a non-zero polynomial } P(z_1, \ldots, z_N): \]

1. **Usefulness:** \( \forall f \in C, P(\text{coeff}(f)) = 0. \)
2. **Constructivity:** \( P \) has degree \( \text{poly}(N) \) and size \( \text{poly}(N) \).
3. **Largeness:** \( P(\text{coeff}(h)) \neq 0 \) for candidate hard polynomial \( h \) (and for many more polynomials).

Example 1: \( C = \{(\alpha x - \beta)^2 \mid \alpha, \beta \in \mathbb{C}\} \). Then, \( P(z_1, z_2, z_3) = z_2^2 - 4z_1z_3 \) such that \( P(\text{coeff}(f)) = 0 \) for all \( f \in C \).

Example 2: \( C = \{\Sigma \Pi \Sigma, \Sigma \Pi \Sigma \Pi, \Sigma \land \Sigma\} \).
Then, \( P(z_1, \ldots, z_N) = \det(W) \) such that \( P(\text{coeff}(f)) = 0 \) for all \( f \in C \).
How far can natural proofs succeed?

Can we prove Valiant’s Conjecture via natural proofs? VP(n) is class of $n$-variate degree-poly(n) polynomials computable by size poly(n) circuits.
How far can natural proofs succeed?

Can we prove Valiant’s Conjecture via natural proofs? VP(\(n\)) is class of \(n\)-variate degree-poly(\(n\)) polynomials computable by size poly(\(n\)) circuits.

**Question:** Does there exist a non-zero polynomial \(P(z_1, \ldots, z_N)\):

- \(\forall f \in \text{VP}(n), P(\text{coeff}(f)) = 0;\)
- \(P\) has degree poly(\(N\)) and size poly(\(N\))?
How far can natural proofs succeed?

Can we prove Valiant’s Conjecture via natural proofs? VP($n$) is class of $n$-variate degree-poly($n$) polynomials computable by size poly($n$) circuits.

**Question:** Does there exist a non-zero polynomial $P(z_1, \ldots, z_N)$:

- $\forall f \in \text{VP}(n), P(\text{coeff}(f)) = 0$;
- $P$ has degree poly($N$) and size poly($N$)?

**Theorem (Chatterjee, Kumar, R., Saptharishi, Tengse)**

**Answer:** Yes, for polynomials with small integer coefficients.
On the Existence of Natural Proofs

Theorem (Chatterjee, Kumar, R., Saptharishi, Tengse)

For $n, d$ and $N = \binom{n+d}{n}$, there exists a non-zero $P(z_1, \ldots, z_N)$ such that

1. $P(\text{coeff}(f)) = 0$ for all $f \in \text{VP}(n, d)$ with small integer coefficients;
2. $P(z_1, \ldots, z_N)$ has size and degree $\text{poly}(N)$; and
3. there exists $h$ having small integer coefficients with $P(\text{coeff}(h)) \neq 0$.

What does this result suggest? An evidence for the power of natural lower bound techniques for proving super-polynomial lower bounds.
On the Existence of Natural Proofs

**Theorem (Chatterjee, Kumar, R., Saptharishi, Tengse)**

For $n, d$ and $N = \binom{n+d}{n}$, there exists a non-zero $P(z_1, \ldots, z_N)$ such that

1. $P(\text{coeff}(f)) = 0$ for all $f \in \text{VP}(n, d)$ with small integer coefficients;
2. $P(z_1, \ldots, z_N)$ has size and degree $\text{poly}(N)$; and
3. there exists $h$ having small integer coefficients with $P(\text{coeff}(h)) \neq 0$.

- What does this result suggest? An evidence for the power of natural lower bound techniques for proving super-polynomial lower bounds.
Strassen’s Vermeidung von Divisionen
Strassen’s Vermeidung von Divisionen
Strassen’s Vermeidung von Divisionen
Strassen’s Vermeidung von Divisionen

\[ \text{ckt \ } \Phi_f \text{ for } f \text{ with } \{+ , \times , \div \} \]
\[ \text{size } s \]

\[ \text{ckt \ } \Phi_f \text{ for } f \text{ with } \{+ , \times , \sqrt[2]{\text{one root}}, \div \} \text{ gate} \]
\[ \text{size } \text{poly}(s) \]
Strassen’s Vermeidung von Divisionen

\[ f = \frac{h}{g} = \frac{h}{1-(1-g)} = h \sum_{j=0}^{\infty} (1-g)^j \]

\( n \)-variable
\( \deg d \)

Can truncate power series depending on the degree of \( f \).

Need \( \overline{a} \in \mathbb{F}^n \) s.t. \( g(\overline{a}) \neq 0 \).
Strassen’s Vermeidung von Divisionen

Division gates can be eliminated with polynomial blow up in size.

\[ f = \frac{h}{g} = \frac{h}{1-(1-g)} = h \sum_{j=0}^{\infty} (1-g)^j \]

\( \eta \)-variate
deg \( d \)

Can truncate power series depending on the degree of \( f \).

Need \( \overline{a} \in F^n \) s.t. \( g(\overline{a}) \neq 0 \).
Polynomial Identity Testing

- A polynomial \( f \equiv 0 \) is \textit{identically zero} if all its coefficients are zero.
- E.g.: \((x + y)^2 - x^2 - y^2 - 2xy \equiv 0\) and \((x + y)^2 - x^2 - y^2 + 2xy \not\equiv 0\).
A polynomial \( f \equiv 0 \) is \textit{identically zero} if all its coefficients are zero.

E.g.: \((x + y)^2 - x^2 - y^2 - 2xy \equiv 0\) and \((x + y)^2 - x^2 - y^2 + 2xy \not\equiv 0\).

**Polynomial Identity Testing (PIT)**

Given \( f \in \mathbb{F}[x_1, \ldots, x_n] \) test if \( f \equiv 0 \).
Polynomial Identity Testing

- A polynomial \( f \equiv 0 \) is *identically zero* if all its coefficients are zero.
- E.g.: \((x + y)^2 - x^2 - y^2 - 2xy \equiv 0\) and \((x + y)^2 - x^2 - y^2 + 2xy \not\equiv 0\).

**Polynomial Identity Testing (PIT)**

Given \( f \in \mathbb{F}[x_1, \ldots, x_n] \) test if \( f \equiv 0 \).

- Univariate case:
A polynomial \((f \equiv 0)\) is *identically zero* if all its coefficients are zero.

E.g.: \((x + y)^2 - x^2 - y^2 - 2xy \equiv 0\) and \((x + y)^2 - x^2 - y^2 + 2xy \not\equiv 0\).

**Polynomial Identity Testing (PIT)**

Given \(f \in \mathbb{F}[x_1, \ldots, x_n]\) test if \(f \equiv 0\).

- **Univariate case:** Any non-zero univariate polynomial of degree \(d\) has at most \(d\) roots. Easy to get a polynomial time algorithm.
Polynomial Identity Testing

- A polynomial \( f \equiv 0 \) is *identically zero* if all its coefficients are zero.
- E.g.: \((x + y)^2 - x^2 - y^2 - 2xy \equiv 0\) and \((x + y)^2 - x^2 - y^2 + 2xy \not\equiv 0\).

**Polynomial Identity Testing (PIT)**

Given \( f \in \mathbb{F}[x_1, \ldots, x_n] \) test if \( f \equiv 0 \).

- **Univariate case:** Any non-zero univariate polynomial of degree \( d \) has at most \( d \) roots. Easy to get a polynomial time algorithm.
- **Multivariate case:** Can have infinitely many roots.
Polynomial Identity Testing

• A polynomial \((f \equiv 0)\) is *identically zero* if all its coefficients are zero.
• E.g.: \((x + y)^2 - x^2 - y^2 - 2xy \equiv 0\) and \((x + y)^2 - x^2 - y^2 + 2xy \not\equiv 0\).

Polynomial Identity Testing (PIT)

Given \(f \in \mathbb{F}[x_1, \ldots, x_n]\) test if \(f \equiv 0\).

• Univariate case: Any non-zero univariate polynomial of degree \(d\) has at most \(d\) roots. Easy to get a polynomial time algorithm.
• Multivariate case: Can have infinitely many roots.
• Randomized polynomial time algorithm for multivariate PIT is known.
• **Open Question:** Derandomizing PIT.
Non-commutative PIT

- Set of non-commuting variables \( \{x_1, \ldots, x_n\} \) i.e., \( x_i x_j \neq x_j x_i \) \( \forall i \neq j \). E.g., \((x_1 + x_2)(x_1 - x_2) \neq x_1^2 - x_2^2\).
- A non-commutative polynomial \( f(x_1, \ldots, x_n) \in \mathbb{F}\langle x_1, \ldots, x_n \rangle \) is a combination of words.
Non-commutative PIT

- Set of non-commuting variables \( \{x_1, \ldots, x_n\} \) i.e., \( x_ix_j \neq x_jx_i \ \forall i \neq j \).
  E.g., \((x_1 + x_2)(x_1 - x_2) \neq x_1^2 - x_2^2\).

- A non-commutative polynomial \( f(x_1, \ldots, x_n) \in \mathbb{F}\langle x_1, \ldots, x_n \rangle \) is a combination of words.

- ncPIT: Given a non-commutative polynomial \( f \in \mathbb{F}\langle x_1, \ldots, x_n \rangle \) test if \( f \equiv 0 \). E.g., \( x_1x_2 - x_2x_1 \not\equiv 0 \) in the non-commutative world.
Non-commutative PIT

- Set of non-commuting variables \( \{x_1, \ldots, x_n\} \) i.e., \( x_i x_j \neq x_j x_i \forall i \neq j \). E.g., \( (x_1 + x_2)(x_1 - x_2) \neq x_1^2 - x_2^2 \).

- A non-commutative polynomial \( f(x_1, \ldots, x_n) \in \mathbb{F}\langle x_1, \ldots, x_n \rangle \) is a combination of words.

- \textbf{ncPIT: Given} a non-commutative polynomial \( f \in \mathbb{F}\langle x_1, \ldots, x_n \rangle \) test if \( f \equiv 0 \). E.g., \( x_1 x_2 - x_2 x_1 \not\equiv 0 \) in the non-commutative world.

- Non-commutative circuit: arithmetic circuit whose \( \times \) gate respects the ordering.
Non-commutative PIT

- Set of non-commuting variables \( \{x_1, \ldots, x_n\} \) i.e., \( x_i x_j \neq x_j x_i \ \forall i \neq j \). E.g., \( (x_1 + x_2)(x_1 - x_2) \neq x_1^2 - x_2^2 \).

- A non-commutative polynomial \( f(x_1, \ldots, x_n) \in \mathbb{F}\langle x_1, \ldots, x_n \rangle \) is a combination of words.

- **ncPIT:** Given a non-commutative polynomial \( f \in \mathbb{F}\langle x_1, \ldots, x_n \rangle \) test if \( f \equiv 0 \). E.g., \( x_1 x_2 - x_2 x_1 \not\equiv 0 \) in the non-commutative world.

- Non-commutative circuit: arithmetic circuit whose \( \times \) gate respects the ordering.

- **(Amitsur-Levitski ‘50)** Let \( f(x_1, \ldots, x_n) \in \mathbb{F}\langle x_1, \ldots, x_n \rangle \) be non-zero polynomial of degree \( \leq 2d - 1 \). Then, there exists \( (A_1, \ldots, A_n) \in \text{Mat}_d^n(\mathbb{F}) \) such that \( f(A_1, \ldots, A_n) \neq 0 \) as a matrix.
Non-commutative PIT

- Set of non-commuting variables \( \{x_1, \ldots, x_n\} \) i.e., \( x_i x_j \neq x_j x_i \) \( \forall i \neq j \).
  
  E.g., \( (x_1 + x_2)(x_1 - x_2) \neq x_1^2 - x_2^2 \).

- A non-commutative polynomial \( f(x_1, \ldots, x_n) \in \mathbb{F}\langle x_1, \ldots, x_n \rangle \) is a combination of words.

- \textbf{ncPIT: Given} a non-commutative polynomial \( f \in \mathbb{F}\langle x_1, \ldots, x_n \rangle \) test if \( f \equiv 0 \). E.g., \( x_1 x_2 - x_2 x_1 \not\equiv 0 \) in the non-commutative world.

- Non-commutative circuit: arithmetic circuit whose \( \times \) gate respects the ordering.

- \textbf{(Amitsur-Levitski ‘50)} Let \( f(x_1, \ldots, x_n) \in \mathbb{F}\langle x_1, \ldots, x_n \rangle \) be non-zero polynomial of degree \( \leq 2d - 1 \). Then, there exists \( (A_1, \ldots, A_n) \in \text{Mat}_d^n(\mathbb{F}) \) such that \( f(A_1, \ldots, A_n) \neq 0 \) as a matrix.

- \textbf{(Bogdanov, Wee 2005)} Randomized polynomial time algorithm for ncPIT on circuits with polynomial degree.
Non-commutative PIT

- Set of non-commuting variables \( \{x_1, \ldots, x_n\} \) i.e., \( x_i x_j \neq x_j x_i \forall i \neq j \).
  E.g., \((x_1 + x_2)(x_1 - x_2) \neq x_1^2 - x_2^2\).

- A non-commutative polynomial \( f(x_1, \ldots, x_n) \in \mathbb{F}\langle x_1, \ldots, x_n \rangle \) is a combination of words.

- ncPIT: Given a non-commutative polynomial \( f \in \mathbb{F}\langle x_1, \ldots, x_n \rangle \) test if \( f \equiv 0 \). E.g., \( x_1 x_2 - x_2 x_1 \neq 0 \) in the non-commutative world.

- Non-commutative circuit: arithmetic circuit whose \( \times \) gate respects the ordering.

- (Amitsur-Levitski ‘50) Let \( f(x_1, \ldots, x_n) \in \mathbb{F}\langle x_1, \ldots, x_n \rangle \) be non-zero polynomial of degree \( \leq 2d - 1 \). Then, there exists \( (A_1, \ldots, A_n) \in \text{Mat}^n_d(\mathbb{F}) \) such that \( f(A_1, \ldots, A_n) \neq 0 \) as a matrix.

- (Bogdanov, Wee 2005) Randomized polynomial time algorithm for ncPIT on circuits with polynomial degree.

- Open: Randomized polynomial time algorithm for ncPIT on circuits of polynomial size.
Non-Commutative circuits with division

- Circuits with +, × (ordered multiplication gates) and INV gates. An INV gate has one input and computes $g^{-1}$ on input $g$. 
Non-Commutative circuits with division

- Circuits with $+$, $\times$ (ordered multiplication gates) and INV gates. An INV gate has one input and computes $g^{-1}$ on input $g$.

\[ z \triangleq xy \]
\[ x^{-1} = yz^{-1} \]
\[ y^{-1} = z^{-1}x \]
Non-Commutative circuits with division

- Circuits with $+$, $\times$ (ordered multiplication gates) and INV gates. An INV gate has one input and computes $g^{-1}$ on input $g$.

- Hua’s identity: $(x + xy^{-1}x)^{-1} \equiv x^{-1} + (x + y)^{-1}$.
Non-Commutative circuits with division

- Circuits with $+$, $\times$ (ordered multiplication gates) and INV gates. An INV gate has one input and computes $g^{-1}$ on input $g$.

- Hua’s identity: $(x + xy^{-1}x)^{-1} \equiv x^{-1} + (x + y)^{-1}$.

- Nested inversions cannot always be eliminated. e.g., $(u + xy^{-1}z)^{-1}$.

- Inversion height: number if nested inversions.
Rational Identity Testing

- A rational expression \( r(x_1, \ldots, x_n) \) computes the zero function\(^1\) if
  - \( r \) has a nonempty domain of definition
  - for each \( d \in \mathbb{N} \) and substitution \((A, \ldots, A_n) \in (\text{Mat}_d(\mathbb{F}))^n, r(A, \ldots, A_n) = 0 \) as a matrix when defined.

---

\(^1\)(zero function in the free skew-field)

\(^2\)formula is a circuit whose underlying graph is a tree.
Rational Identity Testing

- A rational expression $r(x_1, \ldots, x_n)$ computes the zero function\(^1\) if
  - $r$ has a nonempty domain of definition
  - for each $d \in \mathbb{N}$ and substitution $(A, \ldots, A_n) \in (\text{Mat}_d(\mathbb{F}))^n$, $r(A, \ldots, A_n) = 0$ as a matrix when defined.

- RIT: Given a non-commutative circuit (with inverses) computing $r$ decide if $r \equiv 0$.

\(^1\)(zero function in the free skew-field)

\(^2\)formula is a circuit whose underlying graph is a tree.
Rational Identity Testing

- A rational expression $r(x_1, \ldots, x_n)$ computes the zero function\(^1\) if:
  - $r$ has a nonempty domain of definition
  - for each $d \in \mathbb{N}$ and substitution $(A, \ldots, A_n) \in (\text{Mat}_d(\mathbb{F}))^n$,
    $r(A, \ldots, A_n) = 0$ as a matrix when defined.

- **RIT:** Given a non-commutative circuit (with inverses) computing $r$ decide if $r \equiv 0$.

- **Open:** Subexponential-time randomized white-box algorithm for noncommutative circuits.

---

\(^1\)(zero function in the free skew-field)

\(^2\)formula is a circuit whose underlying graph is a tree.
Rational Identity Testing

- A rational expression $r(x_1, \ldots, x_n)$ computes the zero function\(^1\) if
  - $r$ has a nonempty domain of definition
  - for each $d \in \mathbb{N}$ and substitution $(A, \ldots, A_n) \in (\text{Mat}_d(\mathbb{F}))^n$, $r(A, \ldots, A_n) = 0$ as a matrix when defined.

- **RIT:** Given a non-commutative circuit (with inverses) computing $r$, decide if $r \equiv 0$.

- **Open:** Subexponential-time randomized white-box algorithm for noncommutative circuits.

  (Garg et al. ‘20, Ivanyos et al. ‘18) Deterministic polynomial time algorithm in the white-box model for non-commutative formula\(^2\).

  (Derksen, Makam ‘17) Randomized polynomial time in black-box model for non-commutative formula.

\(^1\)(zero function in the free skew-field)

\(^2\)Formula is a circuit whose underlying graph is a tree.
Lower Bounds $\implies$ RIT algorithm

- A polynomial identity for $d \times d$ matrix algebra is a noncommutative polynomial $p(x_1, \ldots, x_n)$ that vanishes on $d \times d$ matrix substitutions.
Lower Bounds $\iff$ RIT algorithm

- A polynomial identity for $d \times d$ matrix algebra is a noncommutative polynomial $p(x_1, \ldots, x_n)$ that vanishes on $d \times d$ matrix substitutions.
- $s(x_1, \ldots, x_{2d}) = \sum_{\sigma} \text{sgn}(\sigma)x_{\sigma(1)} \cdots x_{\sigma(2d)}$ is a polynomial identity for $\mathbb{F}^{d \times d}$.
Lower Bounds $\implies$ RIT algorithm

- A polynomial identity for $d \times d$ matrix algebra is a noncommutative polynomial $p(x_1, \ldots, x_n)$ that vanishes on $d \times d$ matrix substitutions.
- $s(x_1, \ldots, x_{2d}) = \sum_{\sigma} \text{sgn}(\sigma)x_{\sigma(1)} \cdots x_{\sigma(2d)}$ is a polynomial identity for $\mathbb{F}^{d \times d}$.

Conjecture (Bogdanov, Wee '05): The minimum size of a branching program of a polynomial identity for the $d \times d$ matrix algebra is $2^{\Omega(d)}$.
Lower Bounds $\implies$ RIT algorithm

- **A polynomial identity** for $d \times d$ matrix algebra is a noncommutative polynomial $p(x_1, \ldots, x_n)$ that vanishes on $d \times d$ matrix substitutions.
- $s(x_1, \ldots, x_{2d}) = \sum_{\sigma} \text{sgn}(\sigma)x_{\sigma(1)} \cdots x_{\sigma(2d)}$ is a polynomial identity for $F^{d \times d}$.

**Conjecture** (Bogdanov, Wee '05): The minimum size of a branching program of a polynomial identity for the $d \times d$ matrix algebra is $2^{\Omega(d)}$.

**Theorem** (Arvind, Chatterjee, Ghosal, Mukhopadhyay, R.,)

If BW conjecture is true then there is a deterministic subexponential time blackbox RIT algorithm for rational formulas of size $s$ over $n$ variables and inversion height $\approx \frac{\log s}{\log \log s}$.
Lower Bounds $\implies$ RIT algorithm

- A polynomial identity for $d \times d$ matrix algebra is a noncommutative polynomial $p(x_1, \ldots, x_n)$ that vanishes on $d \times d$ matrix substitutions.
- $s(x_1, \ldots, x_{2d}) = \sum_{\sigma} \text{sgn}(\sigma) x_{\sigma(1)} \cdots x_{\sigma(2d)}$ is a polynomial identity for $\mathbb{F}^{d \times d}$.

Conjecture (Bogdanov, Wee ‘05): The minimum size of a branching program of a polynomial identity for the $d \times d$ matrix algebra is $2^{\Omega(d)}$.

Theorem (Arvind, Chatterjee, Ghosal, Mukhopadhyay, R.,)

If BW conjecture is true then there is a deterministic subexponential time blackbox RIT algorithm for rational formulas of of size $s$ over $n$ variables and inversion height $\approx \frac{\log s}{\log \log s}$.

(Hrubes, Wigderson) A rational formula of size $s$ has inversion height $O(\log s)$.
Thank you