Lower Bounds for some Algebraic Models of Computation

Prerona Chatterjee

March 27, 2024

Complexity Theory

Q: Given a computational problem and constraints on the computational power at hand,

Complexity Theory

Q: Given a computational problem and constraints on the computational power at hand,

- design a computational model that captures the constraints

Complexity Theory

Q: Given a computational problem and constraints on the computational power at hand,

- design a computational model that captures the constraints
- study the amount of resource required by the model to complete the task.

Complexity Theory

Q: Given a computational problem and constraints on the computational power at hand,

- design a computational model that captures the constraints
- study the amount of resource required by the model to complete the task.

Traditional Time Complexity: Given a boolean function f on n inputs, how many steps are required by a Turing machine to compute the f (in terms of n)?

Complexity Theory

Q: Given a computational problem and constraints on the computational power at hand,

- design a computational model that captures the constraints
- study the amount of resource required by the model to complete the task.

Traditional Time Complexity: Given a boolean function f on n inputs, how many steps are required by a Turing machine to compute the f (in terms of n)?

Communication Complexity: Given a boolean function f, assuming Alice and Bob are computationally unbounded but have partial inputs $\mathbf{x}, \mathbf{y} \in \{0,1\}^n$ respectively, how many bits need to be communicated for them to know $f(\mathbf{x}, \mathbf{y})$ (in terms of n)?

Complexity Theory

Q: Given a computational problem and constraints on the computational power at hand,

- design a computational model that captures the constraints
- study the amount of resource required by the model to complete the task.

Traditional Time Complexity: Given a boolean function f on n inputs, how many steps are required by a Turing machine to compute the f (in terms of n)?

Communication Complexity: Given a boolean function f, assuming Alice and Bob are computationally unbounded but have partial inputs $\mathbf{x}, \mathbf{y} \in \{0,1\}^n$ respectively, how many bits need to be communicated for them to know $f(\mathbf{x}, \mathbf{y})$ (in terms of n)?

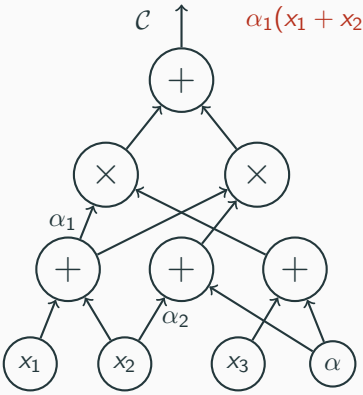
Boolean Circuit Complexity: Given a boolean function f on n inputs, how many $\wedge$, $\vee$, $\neg$ gates are needed for a boolean circuit to compute f (in terms of n)?

Computing Formal Polynomials: Algebraic Models of Computation

Q: Given $f(\mathbf{x}) \in \mathbb{F}[x_1, \ldots, x_n]$ of degree d, how many $+, \times, -$ gates are needed to compute f?

Computing Formal Polynomials: Algebraic Models of Computation

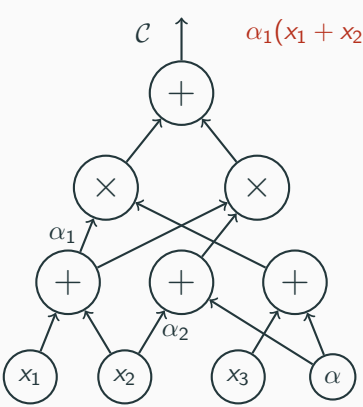
Q: Given $f(\mathbf{x}) \in \mathbb{F}[x_1, \ldots, x_n]$ of degree d, how many $+, \times, -$ gates are needed to compute f?



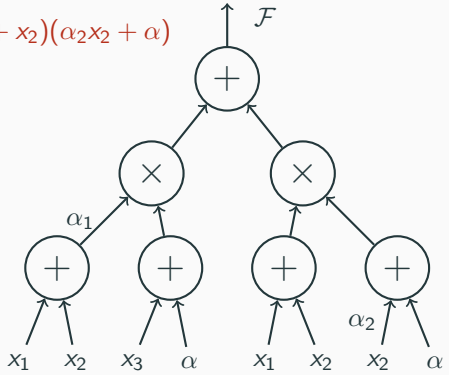
$$\alpha_1(x_1 + x_2)(x_3 + \alpha) + (x_1 + x_2)(\alpha_2 x_2 + \alpha)$$

Computing Formal Polynomials: Algebraic Models of Computation

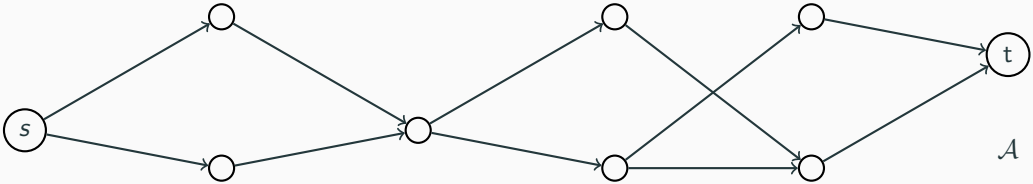
Q: Given $f(\mathbf{x}) \in \mathbb{F}[x_1, \ldots, x_n]$ of degree d, how many $+, \times, -$ gates are needed to compute f?



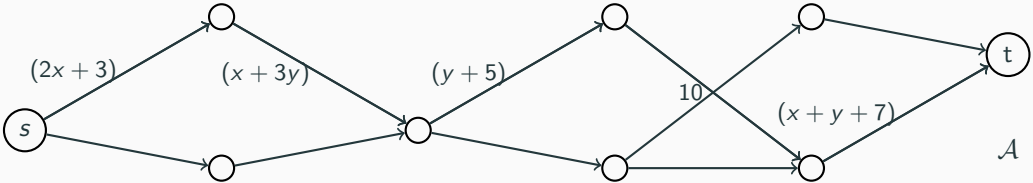
$$\alpha_1(x_1 + x_2)(x_3 + \alpha) + (x_1 + x_2)(\alpha_2 x_2 + \alpha)$$



Algebraic Branching Programs

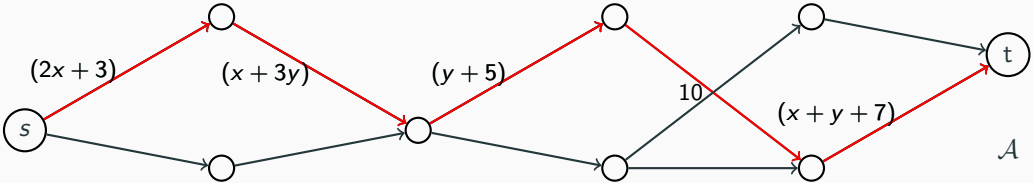


Algebraic Branching Programs



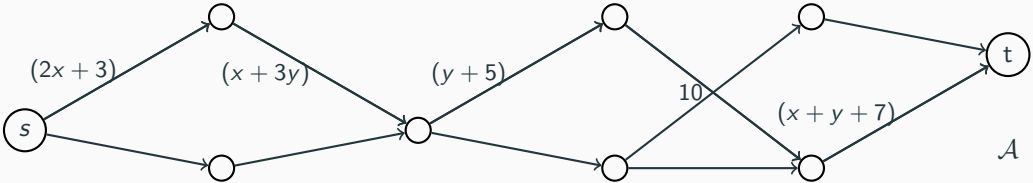
- Label on each edge: An affine linear form in $\{x_1, x_2, \ldots, x_n\}$

Algebraic Branching Programs



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

Algebraic Branching Programs



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

Lower Bounds in Algebraic Circuit Complexity

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

Lower Bounds in Algebraic Circuit Complexity

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

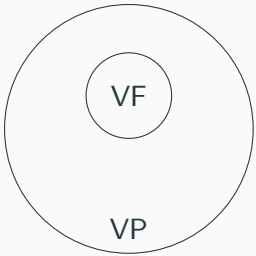
VP: Polynomials computable by circuits of size $\text{poly}(n, d)$.

Lower Bounds in Algebraic Circuit Complexity

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

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

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



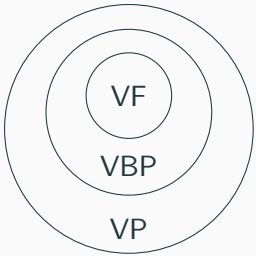
Lower Bounds in Algebraic Circuit Complexity

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

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

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

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



Lower Bounds in Algebraic Circuit Complexity

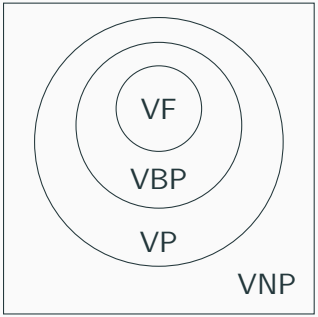
Objects of Study: Polynomials over n variables of degree d.

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

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

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

VNP: Explicit Polynomials



Lower Bounds in Algebraic Circuit Complexity

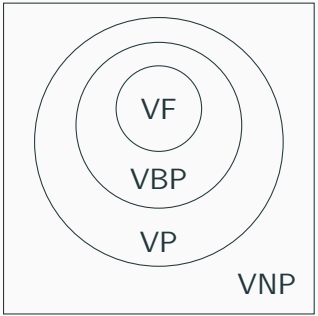
Objects of Study: Polynomials over n variables of degree d.

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

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

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

VNP: Explicit Polynomials



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

Lower Bounds in Algebraic Circuit Complexity

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

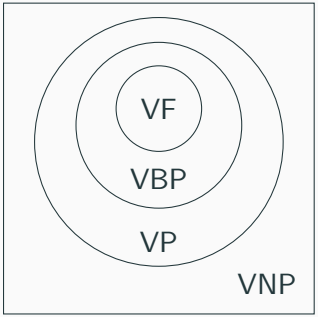
VF: Polynomials computable by formulas of size poly(n, d).

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

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

VNP: Explicit Polynomials

Are the inclusions tight?



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

Lower Bounds for General Models

General Circuits

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

Lower Bounds for General Models

General Circuits

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

General ABPs

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

Lower Bounds for General Models

General Circuits

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

General ABPs

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

General Formulas

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

Lower Bounds for General Models

General Circuits

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

General ABPs

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

General Formulas

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

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

Lower Bounds for General Models

General Circuits

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

General ABPs

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

General Formulas

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

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

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

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

How does one make progress?

Structural Results

Show that if a structured n-variate, degree-d polynomial is computable by a general model of size s, then they can also be computed by a structured model of size $\mathrm{func}(s, n, d)$ for some function func.

How does one make progress?

Structural Results

Show that if a structured n-variate, degree-d polynomial is computable by a general model of size s, then they can also be computed by a structured model of size $\mathrm{func}(s, n, d)$ for some function func.

Study Structured Models

Prove strong lower bounds against structured models computing f.

How does one make progress?

Structural Results

Show that if a structured n-variate, degree-d polynomial is computable by a general model of size s, then they can also be computed by a structured model of size $\text{func}(s, n, d)$ for some function func.

[Agrawal-Vinay, Koiran, Tavenas]

Size s circuits computing n-variate degree d polynomials can be converted into depth-4 circuits of size $s^{O(\sqrt{d})}$.

Study Structured Models

Prove strong lower bounds against structured models computing f.

How does one make progress?

Structural Results

Show that if a structured n-variate, degree-d polynomial is computable by a general model of size s, then they can also be computed by a structured model of size $\text{func}(s, n, d)$ for some function func.

[Agrawal-Vinay, Koiran, Tavenas]

Size s circuits computing n-variate degree d polynomials can be converted into depth-4 circuits of size $s^{O(\sqrt{d})}$.

[Gupta-Kamath-Kayal-Saptharishi]

Size s circuits computing n-variate degree d polynomials can be converted into depth-3 circuits of size $s^{O(\sqrt{d})}$.

Study Structured Models

Prove strong lower bounds against structured models computing f.

How does one make progress?

Structural Results

Show that if a structured n-variate, degree-d polynomial is computable by a general model of size s, then they can also be computed by a structured model of size $\text{func}(s, n, d)$ for some function func.

[Agrawal-Vinay, Koiran, Tavenas]

Size s circuits computing n-variate degree d polynomials can be converted into depth-4 circuits of size $s^{O(\sqrt{d})}$.

[Gupta-Kamath-Kayal-Saptharishi]

Size s circuits computing n-variate degree d polynomials can be converted into depth-3 circuits of size $s^{O(\sqrt{d})}$.

Study Structured Models

Prove strong lower bounds against structured models computing f.

A lot of work that culminated in

[Limaye-Srinivasan-Tavenas]

Any depth-3 or depth-4 circuit computing $\text{IMM}_{n,\log n}(\mathbf{x})$ must have size $n^{\Omega(\sqrt{d})}$.

Towards Better ABP Lower Bounds

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

Towards Better ABP Lower Bounds

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

[Bhargav-Dwivedi-Saxena]: Super polynomial lower bound against total-width of $\sum$osmABP for a polynomial of degree $d = O\left(\frac{\log n}{\log \log n}\right)$ $\implies$ super-polynomial lower bound against ABPs.

Towards Better ABP Lower Bounds

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

[Bhargav-Dwivedi-Saxena]: Super polynomial lower bound against total-width of $\sum$osmABP for a polynomial of degree $d = O\left(\frac{\log n}{\log \log n}\right)$ $\implies$ super-polynomial lower bound against ABPs.

[C-Kush-Saraf-Shpilka]: For $\omega(\log n) = d \leq n$, there is a polynomial $G_{n,d}(\mathbf{x})$ which is set-multilinear w.r.t $\mathbf{x} = \{\mathbf{x}_1, \ldots, \mathbf{x}_d\}$, where $|\mathbf{x}_i| \leq n$ for every $i \in [d]$, such that:

- $G_{n,d}$ is computable by a set-multilinear ABP of size poly(n),
- any $\sum$osmABP computing $G_{n,d}$ must have super-polynomial total-width.

Set-Multilinearity

The variable set is divided into buckets.

$$\mathbf{x} = \mathbf{x}_1 \cup \cdots \cup \mathbf{x}_d \quad \text{where} \quad \mathbf{x}_i = \{x_{i,1}, \ldots x_{i,n_i}\}.$$

Set-Multilinearity

The variable set is divided into buckets.

$$\mathbf{x} = \mathbf{x}_1 \cup \cdots \cup \mathbf{x}_d \quad \text{where} \quad \mathbf{x}_i = \{x_{i,1}, \ldots x_{i,n_i}\}.$$

f is set-multilinear with respect to $\{\mathbf{x}_1, \ldots, \mathbf{x}_d\}$ if

every monomial in f has exactly one variable from $\mathbf{x}_i$ for each $i \in [d]$.

Set-Multilinearity

The variable set is divided into buckets.

$$\mathbf{x} = \mathbf{x}_1 \cup \cdots \cup \mathbf{x}_d \quad \text{where} \quad \mathbf{x}_i = \{x_{i,1}, \ldots x_{i,n_i}\}.$$

f is set-multilinear with respect to $\{\mathbf{x}_1, \ldots, \mathbf{x}_d\}$ if

every monomial in f has exactly one variable from $\mathbf{x}_i$ for each $i \in [d]$.

An ABP is set-multilinear with respect to $\{\mathbf{x}_1, \ldots, \mathbf{x}_d\}$ if every path in it

computes a set-multilinear monomial with respect to $\{\mathbf{x}_1, \ldots, \mathbf{x}_d\}$.

Near Tightness of ABP Set-Multilinearisation

For $\sigma \in S_d$, an ABP is σ-ordered set-multilinear with respect to $\{\mathbf{x}_1, \ldots, \mathbf{x}_d\}$ if

- there are d layers in the ABP
- every edge in layer i is labelled by a homogeneous linear form in $\mathbf{x}_{\sigma(i)}$

Near Tightness of ABP Set-Multilinearisation

For $\sigma \in S_d$, an ABP is σ-ordered set-multilinear with respect to $\{\mathbf{x}_1, \ldots, \mathbf{x}_d\}$ if

- there are d layers in the ABP
- every edge in layer i is labelled by a homogeneous linear form in $\mathbf{x}_{\sigma(i)}$

$\sum$osmABP: Sum of ordered set-multilinear ABPs, each with a possibly different ordering.

Near Tightness of ABP Set-Multilinearisation

For $\sigma \in S_d$, an ABP is σ-ordered set-multilinear with respect to $\{\mathbf{x}_1, \ldots, \mathbf{x}_d\}$ if

- there are d layers in the ABP
- every edge in layer i is labelled by a homogeneous linear form in $\mathbf{x}_{\sigma(i)}$

$\sum$osmABP: Sum of ordered set-multilinear ABPs, each with a possibly different ordering.

[Bhargav-Dwivedi-Saxena]: Super polynomial lower bound against total-width of $\sum$osmABP for a polynomial of degree $d = O\left(\frac{\log n}{\log\log n}\right) \implies$ super-polynomial lower bound against ABPs.

Near Tightness of ABP Set-Multilinearisation

For $\sigma \in S_d$, an ABP is σ-ordered set-multilinear with respect to $\{\mathbf{x}_1, \ldots, \mathbf{x}_d\}$ if

- there are d layers in the ABP
- every edge in layer i is labelled by a homogeneous linear form in $\mathbf{x}_{\sigma(i)}$

$\sum$osmABP: Sum of ordered set-multilinear ABPs, each with a possibly different ordering.

[Bhargav-Dwivedi-Saxena]: Super polynomial lower bound against total-width of $\sum$osmABP for a polynomial of degree $d = O\left(\frac{\log n}{\log \log n}\right)$ $\implies$ super-polynomial lower bound against ABPs.

[C-Kush-Saraf-Shpilka]: Super polynomial lower bound against total-width of $\sum$osmABP for a polynomial of degree $d = \omega(\log n)$ that is computable by polynomial-sized ABPs.

Non-Commutativity

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

Non-Commutativity

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

Non-Commutative Models: The multiplication gates, additionally, respect the order.

Non-Commutativity

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

Non-Commutative Models: The multiplication gates, additionally, respect the order.

Can we do better in this setting?

Non-Commutativity

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

Non-Commutative Models: The multiplication gates, additionally, respect the order.

Can we do better in this setting? For general circuits, continues to be $\Omega(n \log d)$.

Non-Commutativity

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

Non-Commutative Models: The multiplication gates, additionally, respect the order.

Can we do better in this setting? For general circuits, continues to be $\Omega(n \log d)$.

[C-Hrubeš]: Any homogeneous non-commutative circuit computing

$$\mathrm{OSym}_{n,d}(\mathbf{x}) = \sum_{1 \leq i_1 < \cdots < i_d \leq n} x_{i_1} \cdots x_{i_d}$$

has size $\Omega(nd)$ for $d \leq \frac{n}{2}$.

Non-Commutativity

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

Non-Commutative Models: The multiplication gates, additionally, respect the order.

Can we do better in this setting? For general circuits, continues to be $\Omega(n \log d)$.

[C-Hrubeš]: Any homogeneous non-commutative circuit computing

$$\mathrm{OSym}_{n,d}(\mathbf{x}) = \sum_{1 \leq i_1 < \cdots < i_d \leq n} x_{i_1} \cdots x_{i_d}$$

has size $\Omega(nd)$ for $d \leq \frac{n}{2}$. The lower bound is tight for homogeneous non-commutative circuits.

Non-Commutativity

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

Non-Commutative Models: The multiplication gates, additionally, respect the order.

Can we do better in this setting? For general circuits, continues to be $\Omega(n \log d)$.

[C-Hrubeš]: Any homogeneous non-commutative circuit computing

$$\mathrm{OSym}_{n,d}(\mathbf{x}) = \sum_{1 \leq i_1 < \cdots < i_d \leq n} x_{i_1} \cdots x_{i_d}$$

has size $\Omega(nd)$ for $d \leq \frac{n}{2}$. The lower bound is tight for homogeneous non-commutative circuits.

[Nisan]: Any ABP computing $\mathrm{Pal}_n(x_0, x_1) = \sum_{w \in \{0,1\}^{n/2}} \mathbf{x}_w \cdot \mathbf{x}_{w^R}$ has size $2^{\Omega(n)}$.

ABP vs Formula in the Non-Commutative Setting

[Tavenas-Limaye-Srinivasan]: Any homogeneous non-commutative formula computing $\mathrm{IMM}_{n,d}(\mathbf{x})$ has size $n^{\Omega(\log \log d)}$.

ABP vs Formula in the Non-Commutative Setting

[Tavenas-Limaye-Srinivasan]: Any homogeneous non-commutative formula computing $\mathrm{IMM}_{n,d}(\mathbf{x})$ has size $n^{\Omega(\log \log d)}$.

(bucket index) ordered set-multilinear $\equiv$ homogeneous non-commutative (position index)

ABP vs Formula in the Non-Commutative Setting

[Tavenas-Limaye-Srinivasan]: Any homogeneous non-commutative formula computing $\mathrm{IMM}_{n,d}(\mathbf{x})$ has size $n^{\Omega(\log\log d)}$.

(bucket index) ordered set-multilinear $\equiv$ homogeneous non-commutative (position index)

Abecedarian Polynomials: Let $f \in \mathbb{F}\langle \mathbf{x} \rangle$ and $\{X_1, \ldots, X_m\}$ be a partition of $\mathbf{x}$ into buckets.

ABP vs Formula in the Non-Commutative Setting

[Tavenas-Limaye-Srinivasan]: Any homogeneous non-commutative formula computing $\mathrm{IMM}_{n,d}(\mathbf{x})$ has size $n^{\Omega(\log\log d)}$.

(bucket index) ordered set-multilinear $\equiv$ homogeneous non-commutative (position index)

Abecedarian Polynomials: Let $f \in \mathbb{F}\langle \mathbf{x} \rangle$ and $\{X_1, \ldots, X_m\}$ be a partition of $\mathbf{x}$ into buckets. f is abecedarian with respect to $\{X_1, \ldots, X_m\}$ if every monomial in f has the form $X_1^* X_2^* \cdots X_m^*$.

ABP vs Formula in the Non-Commutative Setting

[Tavenas-Limaye-Srinivasan]: Any homogeneous non-commutative formula computing $\mathrm{IMM}_{n,d}(\mathbf{x})$ has size $n^{\Omega(\log\log d)}$.

(bucket index) ordered set-multilinear $\equiv$ homogeneous non-commutative (position index)

Abecedarian Polynomials: Let $f \in \mathbb{F}\langle \mathbf{x} \rangle$ and $\{X_1, \ldots, X_m\}$ be a partition of $\mathbf{x}$ into buckets. f is abecedarian with respect to $\{X_1, \ldots, X_m\}$ if every monomial in f has the form $X_1^* X_2^* \cdots X_m^*$.

[**Cha**]: For $\mathbf{x} = \cup_{i\in[n]} \{X_i\}$ with $X_i = \{x_{i,j}\}_{j\in[n]}$,

ABP vs Formula in the Non-Commutative Setting

[Tavenas-Limaye-Srinivasan]: Any homogeneous non-commutative formula computing $\mathrm{IMM}_{n,d}(\mathbf{x})$ has size $n^{\Omega(\log\log d)}$.

(bucket index) ordered set-multilinear $\equiv$ homogeneous non-commutative (position index)

Abecedarian Polynomials: Let $f \in \mathbb{F}\langle \mathbf{x} \rangle$ and $\{X_1, \ldots, X_m\}$ be a partition of $\mathbf{x}$ into buckets. f is abecedarian with respect to $\{X_1, \ldots, X_m\}$ if every monomial in f has the form $X_1^* X_2^* \cdots X_m^*$.

[**Cha**]: For $\mathbf{x} = \cup_{i\in[n]} \{X_i\}$ with $X_i = \{x_{i,j}\}_{j\in[n]}$, there exists a $(\log n)$-degree abecedarian polynomial $f \in \mathbb{F}\langle \mathbf{x} \rangle$

ABP vs Formula in the Non-Commutative Setting

[Tavenas-Limaye-Srinivasan]: Any homogeneous non-commutative formula computing $\mathrm{IMM}_{n,d}(\mathbf{x})$ has size $n^{\Omega(\log\log d)}$.

(bucket index) ordered set-multilinear $\equiv$ homogeneous non-commutative (position index)

Abecedarian Polynomials: Let $f \in \mathbb{F}\langle \mathbf{x} \rangle$ and $\{X_1, \ldots, X_m\}$ be a partition of $\mathbf{x}$ into buckets. f is abecedarian with respect to $\{X_1, \ldots, X_m\}$ if every monomial in f has the form $X_1^* X_2^* \cdots X_m^*$.

[**Cha**]: For $\mathbf{x} = \cup_{i\in[n]} \{X_i\}$ with $X_i = \{x_{i,j}\}_{j\in[n]}$, there exists a $(\log n)$-degree abecedarian polynomial $f \in \mathbb{F}\langle \mathbf{x} \rangle$ such that any abecedarian formula computing it has size $n^{\Omega(\log\log n)}$.

ABP vs Formula in the Non-Commutative Setting

[Tavenas-Limaye-Srinivasan]: Any homogeneous non-commutative formula computing $\mathrm{IMM}_{n,d}(\mathbf{x})$ has size $n^{\Omega(\log\log d)}$.

(bucket index) ordered set-multilinear $\equiv$ homogeneous non-commutative (position index)

Abecedarian Polynomials: Let $f \in \mathbb{F}\langle \mathbf{x} \rangle$ and $\{X_1, \ldots, X_m\}$ be a partition of $\mathbf{x}$ into buckets. f is abecedarian with respect to $\{X_1, \ldots, X_m\}$ if every monomial in f has the form $X_1^* X_2^* \cdots X_m^*$.

[**Cha**]: For $\mathbf{x} = \cup_{i\in[n]} \{X_i\}$ with $X_i = \{x_{i,j}\}_{j\in[n]}$, there exists a $(\log n)$-degree abecedarian polynomial $f \in \mathbb{F}\langle \mathbf{x} \rangle$ such that any abecedarian formula computing it has size $n^{\Omega(\log\log n)}$.

If an n-variate polynomial is abecedarian with respect to $\{X_1, \ldots, X_m\}$ for $m = \log n$,

ABP vs Formula in the Non-Commutative Setting

[Tavenas-Limaye-Srinivasan]: Any homogeneous non-commutative formula computing $\mathrm{IMM}_{n,d}(\mathbf{x})$ has size $n^{\Omega(\log\log d)}$.

(bucket index) ordered set-multilinear $\equiv$ homogeneous non-commutative (position index)

Abecedarian Polynomials: Let $f \in \mathbb{F}\langle \mathbf{x} \rangle$ and $\{X_1, \ldots, X_m\}$ be a partition of $\mathbf{x}$ into buckets. f is abecedarian with respect to $\{X_1, \ldots, X_m\}$ if every monomial in f has the form $X_1^* X_2^* \cdots X_m^*$.

[**Cha**]: For $\mathbf{x} = \cup_{i\in[n]} \{X_i\}$ with $X_i = \{x_{i,j}\}_{j\in[n]}$, there exists a $(\log n)$-degree abecedarian polynomial $f \in \mathbb{F}\langle \mathbf{x} \rangle$ such that any abecedarian formula computing it has size $n^{\Omega(\log\log n)}$.

If an n-variate polynomial is abecedarian with respect to $\{X_1, \ldots, X_m\}$ for $m = \log n$, then any formula computing f can be made abecedarian with only $\mathrm{poly}(n)$ blow-up in size.

Some Open Directions

- Better lower bounds against homogeneous formulas?

Some Open Directions

- Better lower bounds against homogeneous formulas?
- Better lower bounds against set-multilinear ABPs?

Some Open Directions

- Better lower bounds against homogeneous formulas?
- Better lower bounds against set-multilinear ABPs?
- Bootstrapping statement, similar to [CILM], which is sensitive to both degree and number of variables?

Some Open Directions

- Better lower bounds against homogeneous formulas?
- Better lower bounds against set-multilinear ABPs?
- Bootstrapping statement, similar to [CILM], which is sensitive to both degree and number of variables?
- Separating formulas and ABPs in the non-commutative setting?

Some Open Directions

- Better lower bounds against homogeneous formulas?
- Better lower bounds against set-multilinear ABPs?
- Bootstrapping statement, similar to [CILM], which is sensitive to both degree and number of variables?
- Separating formulas and ABPs in the non-commutative setting?

Questions?