Superpolynomial lower bounds for circuits of constant depth

Nutan Limaye, Srikanth Srinivasan, Sébastien Tavenas

26 / 09 / 2023

Spoiler

$P(x_1, \ldots, x_N) = \sum_{S \subseteq [N]} \prod_{i \in S} x_i.$

Spoiler

$P(x_1, \ldots, x_N) = \sum_{S \subseteq [N]} \prod_{i \in S} x_i$. Needs $O(2^N)$ operations.

Spoiler

$P(x_1, \ldots, x_N) = \sum_{S \subseteq [N]} \prod_{i \in S} x_i$. Needs $O(2^N)$ operations.

$P(x_1, \ldots, x_N) = \prod_{i \in [N]} (1 + x_i)$.

Spoiler

$P(x_1, \ldots, x_N) = \sum_{S \subseteq [N]} \prod_{i \in S} x_i$. Needs $O(2^N)$ operations.

$P(x_1, \ldots, x_N) = \prod_{i \in [N]} (1 + x_i)$. Needs $O(N)$ operations.

Spoiler

$P(x_1, \ldots, x_N) = \sum_{S \subseteq [N]} \prod_{i \in S} x_i$. Needs $O(2^N)$ operations.

$P(x_1, \ldots, x_N) = \prod_{i \in [N]} (1 + x_i)$. Needs $O(N)$ operations.

How many operations are needed for computing a polynomial?

Spoiler

$P(x_1, \ldots, x_N) = \sum_{S \subseteq [N]} \prod_{i \in S} x_i$. Needs $O(2^N)$ operations.

$P(x_1, \ldots, x_N) = \prod_{i \in [N]} (1 + x_i)$. Needs $O(N)$ operations.

How many operations are needed for computing a polynomial?

Main result,

$\exists H$ which can not be written of the form:
$H(x_1, \ldots, x_N) = \sum_{i_1 \in [N]} \prod_{i_2 \in [N]} \cdots \sum_{i_{p-1} \in [N]} \prod_{i_p \in [N]} T_{i_1, \ldots, i_p}$
where

- T_i are constants or variables,
- the number of alternations between $\sum$ and $\prod$ is bounded by a constant.

Model(s) for Evaluating a Polynomial

Let $P(x_1, \ldots, x_N) \in \mathbb{F}[x_1, \ldots, x_N]$ (In this talk think $\mathbb{F} = \mathbb{Q}$)

Model(s) for Evaluating a Polynomial

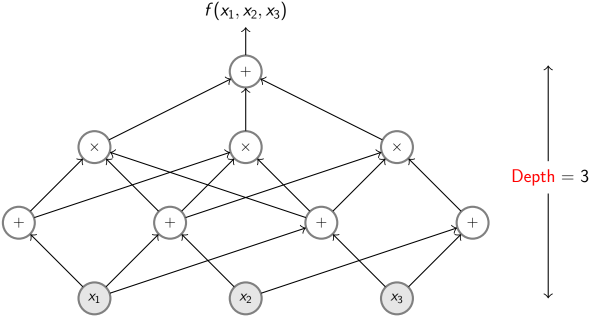
Let $P(x_1, \ldots, x_N) \in \mathbb{F}[x_1, \ldots, x_N]$ (In this talk think $\mathbb{F} = \mathbb{Q}$)

An algebraic circuit (or 'straight-line program') is:

Model(s) for Evaluating a Polynomial

Let $P(x_1, \ldots, x_N) \in \mathbb{F}[x_1, \ldots, x_N]$ (In this talk think $\mathbb{F} = \mathbb{Q}$)

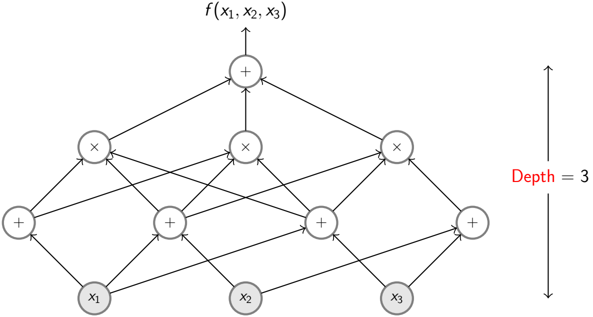
An algebraic circuit (or 'straight-line program') is:



Model(s) for Evaluating a Polynomial

Let $P(x_1, \ldots, x_N) \in \mathbb{F}[x_1, \ldots, x_N]$ (In this talk think $\mathbb{F} = \mathbb{Q}$)

An algebraic circuit (or 'straight-line program') is:

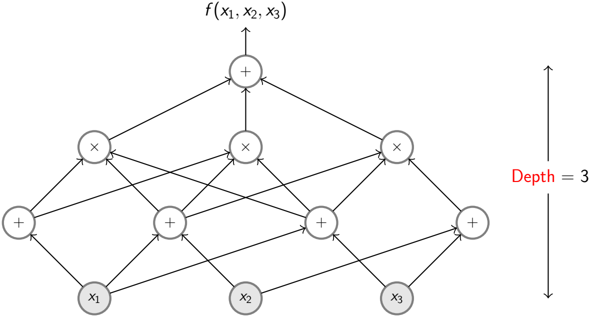


Called $\sum \prod \sum$ circuits.

Model(s) for Evaluating a Polynomial

Let $P(x_1, \ldots, x_N) \in \mathbb{F}[x_1, \ldots, x_N]$ (In this talk think $\mathbb{F} = \mathbb{Q}$)

An algebraic circuit (or 'straight-line program') is:



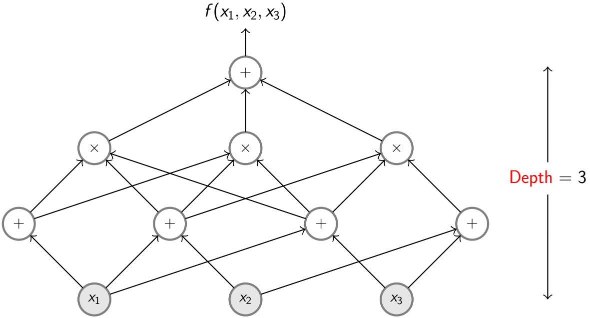
Called $\sum \prod \sum$ circuits.

We will always assume:
the top node is a $\sum$

Model(s) for Evaluating a Polynomial

Let $P(x_1, \ldots, x_N) \in \mathbb{F}[x_1, \ldots, x_N]$ (In this talk think $\mathbb{F} = \mathbb{Q}$)

An algebraic circuit (or 'straight-line program') is:

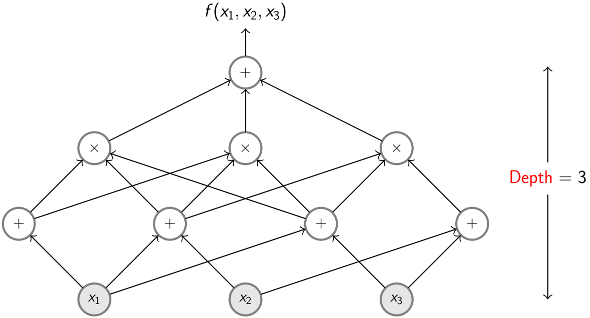


Size $=$ Number of operations.

Model(s) for Evaluating a Polynomial

Let $P(x_1, \ldots, x_N) \in \mathbb{F}[x_1, \ldots, x_N]$ (In this talk think $\mathbb{F} = \mathbb{Q}$)

An algebraic circuit (or 'straight-line program') is:

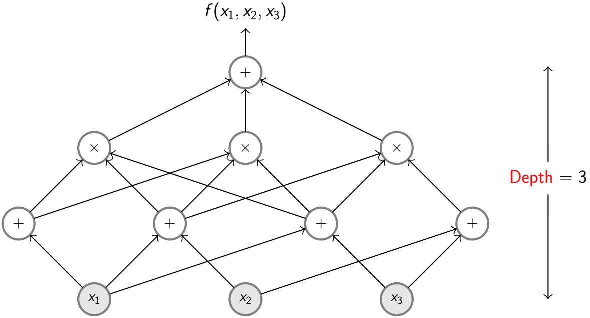


Size = Number of operations. In this case 8.

Model(s) for Evaluating a Polynomial

Let $P(x_1, \ldots, x_N) \in \mathbb{F}[x_1, \ldots, x_N]$ (In this talk think $\mathbb{F} = \mathbb{Q}$)

An algebraic circuit (or ‘straight-line program’) is:



Size = Number of operations. In this case 8.

A formula is a circuit with tree as the underlying undir. graph.

Algebraic analogue of P

Definition (VP – Valiant's P, or "efficiently computable")

Polynomials $f(x_1, \ldots, x_n)$ that can be computed by poly(n)-sized algebraic circuits?

Algebraic analogue of P

Definition (VP – Valiant's P, or "efficiently computable")

Polynomials $f(x_1, \ldots, x_n)$, of degree $d = \text{poly}(n)$, that can be computed by poly(n)-sized algebraic circuits.

In particular they can be "efficiently" simulated by Boolean circuits (bits computation).

Algebraic analogue of P

Definition (VP – Valiant's P, or "efficiently computable")

Polynomials $f(x_1, \ldots, x_n)$, of degree $d = \text{poly}(n)$, that can be computed by $\text{poly}(n)$-sized algebraic circuits.

Examples:

$$\text{[Ben-Or]} \quad \mathrm{ESym}_d(x_1, \cdots, x_n) = \sum_{S \subseteq [n], |S| = d} \prod_{i \in S} x_i$$

$$\text{[Berkowitz,Mahajan-Vinay]} \quad \mathrm{Det}_n = \begin{vmatrix} x_{11} & \cdots & x_{n1} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nn} \end{vmatrix}$$

Algebraic analogue of P

Definition (VP – Valiant's P, or "efficiently computable")

Polynomials $f(x_1, \ldots, x_n)$, of degree $d = \text{poly}(n)$, that can be computed by poly(n)-sized algebraic circuits.

Examples:

[Ben-Or] $$\mathrm{ESym}_d(x_1, \cdots, x_n) = \sum_{S \subseteq [n], |S| = d} \prod_{i \in S} x_i$$

[Berkowitz,Mahajan-Vinay] $$\mathrm{Det}_n = \begin{vmatrix} x_{11} & \cdots & x_{n1} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nn} \end{vmatrix}$$

Fact: [Valiant] Det_n is complete* for VP.

Algebraic analogue of NP

Definition (VNP – Valiant's NP, or "explicit polynomials")

"Anything that can be succinctly described"

Examples:

$$\mathsf{HC}_n$$

$$\mathsf{Perm}_n = \mathrm{perm} \begin{bmatrix} x_{11} & \cdots & x_{n1} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nn} \end{bmatrix} = \sum_{\pi \in S_n} \prod_{i=1}^{n} x_{i\pi(i)}$$

Algebraic analogue of NP

Definition (VNP – Valiant's NP, or "explicit polynomials")

"Anything that can be succinctly described"

An exponential sum of a VP polynomial $g(\mathbf{x}, \mathbf{y})$:

$$f(\mathbf{x}) \quad = \quad \sum_{\mathbf{y} \in \{0,1\}^m} g(\mathbf{x}, \mathbf{y})$$

Examples:

HC_n

$$\mathsf{Perm}_n = \operatorname{perm} \begin{bmatrix} x_{11} & \cdots & x_{n1} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{nn} \end{bmatrix}$$

Algebraic analogue of NP

Definition (VNP – Valiant's NP, or "explicit polynomials")

"Anything that can be succinctly described"

An exponential sum of a VP polynomial $g(\mathbf{x}, \mathbf{y})$:

$$f(\mathbf{x}) \quad = \quad \sum_{\mathbf{y} \in \{0,1\}^m} g(\mathbf{x}, \mathbf{y})$$

Fact: [Valiant] Perm_n is complete for VNP.

Algebraic analogue of NP

Definition (VNP – Valiant's NP, or "explicit polynomials")

"Anything that can be succinctly described"

An exponential sum of a VP polynomial $g(\mathbf{x}, \mathbf{y})$:

$$f(\mathbf{x}) \quad = \quad \sum_{\mathbf{y} \in \{0,1\}^m} g(\mathbf{x}, \mathbf{y})$$

Fact: [Valiant] Perm_n is complete for VNP.

$$\boxed{\text{VP} \quad \text{vs} \quad \text{VNP}}$$

Algebraic analogue of NP

Definition (VNP – Valiant's NP, or "explicit polynomials")

"Anything that can be succinctly described"

An exponential sum of a VP polynomial $g(\mathbf{x}, \mathbf{y})$:

$$f(\mathbf{x}) \quad = \quad \sum_{\mathbf{y} \in \{0,1\}^m} g(\mathbf{x}, \mathbf{y})$$

Fact: [Valiant] Perm_n is complete for VNP.

$$\boxed{\text{VP} \quad \text{vs} \quad \text{VNP} \quad \overset{\sim}{\Longleftrightarrow} \quad \text{Det} \quad \text{vs} \quad \text{Perm}}$$

Algebraic analogue of NP

Definition (VNP – Valiant's NP, or "explicit polynomials")

"Anything that can be succinctly described"

An exponential sum of a VP polynomial $g(\mathbf{x}, \mathbf{y})$:

$$f(\mathbf{x}) \quad = \quad \sum_{\mathbf{y} \in \{0,1\}^m} g(\mathbf{x}, \mathbf{y})$$

Fact: [Valiant] Perm_n is complete for VNP.

$$\boxed{\mathrm{VP} \quad \mathrm{vs} \quad \mathrm{VNP} \quad \overset{\sim}{\Longleftrightarrow} \quad \mathrm{Det} \quad \mathrm{vs} \quad \mathrm{Perm}}$$

Under GRH, $\mathrm{VP} = \mathrm{VNP} \implies \mathrm{P/poly} = \mathrm{NP/poly} = \mathrm{PH/poly}$

How does one begin?

[Baur-Strassen 83]: Any circuit computing $\mathrm{Pow}_n^d = \sum_{i=1}^n X_i^d$ has size at least $\Omega(n \log d)$.

How does one begin?

[Baur-Strassen 83]: Any circuit computing $\mathrm{Pow}_n^d = \sum_{i=1}^n X_i^d$ has size at least $\Omega(n \log d)$.

[Kalorkoti 85]: Any formula computing the polynomial $\sum_{i=1}^n \sum_{j=1}^n X_i^j Y_j$ has size at least $\Omega(n^2)$.

How does one begin?

[Baur-Strassen 83]: Any circuit computing $\mathrm{Pow}_n^d = \sum_{i=1}^n X_i^d$ has size at least $\Omega(n \log d)$.

[Kalorkoti 85]: Any formula computing the polynomial $\sum_{i=1}^n \sum_{j=1}^n X_i^j Y_j$ has size at least $\Omega(n^2)$.

... They are still the best lower bounds for an explicit function!!!!

How does one begin?

[Baur-Strassen 83]: Any circuit computing $\mathrm{Pow}_n^d = \sum_{i=1}^n X_i^d$ has size at least $\Omega(n \log d)$.

[Kalorkoti 85]: Any formula computing the polynomial $\sum_{i=1}^n \sum_{j=1}^n X_i^j Y_j$ has size at least $\Omega(n^2)$.

... They are still the best lower bounds for an explicit function!!!!

"If you can't solve a problem, there is a simpler problem that you can't solve. Find it." – *George Pólya*

Lower bounds against constant depth?

Goal: obtain superpolynomial lower bounds against constant algebraic circuits.

Lower bounds against constant depth?

Goal: obtain superpolynomial lower bounds against constant algebraic circuits.

- Boolean lower bounds imply Algebraic lower bounds

Lower bounds against constant depth?

Goal: obtain superpolynomial lower bounds against constant algebraic circuits.

- Boolean lower bounds imply Algebraic lower bounds
- Exponentially lower bounds for Constant Depth circuits have been known for almost 50 years

Lower bounds against constant depth?

Goal: obtain superpolynomial lower bounds against constant algebraic circuits.

- Boolean lower bounds imply Algebraic lower bounds
- Exponentially lower bounds for Constant Depth circuits have been known for almost 50 years

Lower bounds against constant depth?

Goal: obtain superpolynomial lower bounds against constant algebraic circuits.

- Boolean lower bounds imply Algebraic lower bounds
- Exponentially lower bounds for Constant Depth circuits have been known for almost 50 years
- $\implies$ We can combine them to get our goal!!!

Lower bounds against constant depth?

Goal: obtain superpolynomial lower bounds against constant algebraic circuits.

- Boolean lower bounds imply Algebraic lower bounds
- Exponentially lower bounds for Constant Depth circuits have been known for almost 50 years
- $\Longrightarrow$ We can combine them to get our goal!!!

Problem in the first point:

> - Small Algebraic Circuits simulated by Small Boolean ones
>
> But
>
> - Small Algebraic Circuits of constant depth are not simulated by Small Boolean Circuits of constant depth

See: a sum of variables

Are algebraic constant depth circuits a weak model?

- They can't be simulated by constant-depth Boolean circuits
- $\sum\prod\sum$ can compute $\mathsf{ESym}_{n,d}$ in a non-homogeneous way
- Can simulate general Algebraic Circuits with a subexponential cost!

IMM

Another example of problem in VP: (still almost VP-complete)

$$\underset{A}{\begin{pmatrix} \boxed{} & & \\ & & \\ & & \end{pmatrix}} = \underset{X_1}{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}} \times \underset{X_2}{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}} \times \cdots \times \underset{X_d}{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}}$$

$\mathrm{IMM}_{n,d}$ defined over variable sets $X_1, \ldots, X_d$, each of size n^2.

Each X_i thought of as an $n \times n$ matrix.

$\mathrm{IMM}_{n,d}$ is the $(1,1)$th entry of product $X_1 \cdot X_2 \cdot \ldots \cdot X_d$.
(polynomial with dn^2 variables and degree d)

Reduction to log-depth

Depth to compute $\mathrm{IMM}_{n,d}$?

$$\underbrace{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}}_{X_1} \times \underbrace{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}}_{X_2} \times \quad \cdots \quad \times \underbrace{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}}_{X_d}$$

Reduction to log-depth

Depth to compute $\mathrm{IMM}_{n,d}$?

$$\underbrace{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}}_{X_1} \times \underbrace{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}}_{X_2} \times \quad \cdots \quad \times \underbrace{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}}_{X_d}$$

"Divide and Conquer":

Compute (recursively) $\mathrm{IMM}_{n,d/2}$ on the left and on the right.

Recombine with one matrix multiplication.

Reduction to log-depth

Depth to compute $\mathrm{IMM}_{n,d}$?

$$\underbrace{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}}_{X_1} \times \underbrace{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}}_{X_2} \times \quad \cdots \quad \times \underbrace{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}}_{X_d}$$

"Divide and Conquer":

Compute (recursively) $\mathrm{IMM}_{n,d/2}$ on the left and on the right.

Recombine with one matrix multiplication.

> $\mathrm{IMM}_{n,d}$ is computed by a circuit of size $\mathrm{poly}(n, d)$ and depth $O(\log d)$.

Are algebraic constant depth circuits a weak model?

- They can't be simulated by constant-depth Boolean circuits
- $\sum\prod\sum$ can compute $\mathrm{ESym}_{n,d}$ in a non-homogeneous way
- Can simulate general Algebraic Circuits with a subexponential cost!

[VSBR – AV/K/T – GKKS]

If P can be computed by a circuit of size s,
then it can be computed by a

- log-depth circuit of size $\mathrm{poly}(sN)$,

Reduction to depth 4

Depth-4 circuits for $\mathrm{IMM}_{n,d}$?

$$\underbrace{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}}_{X_1} \times \underbrace{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}}_{X_2} \times \quad \cdots \quad \times \underbrace{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}}_{X_d}$$

Reduction to depth 4

Depth-4 circuits for $\mathrm{IMM}_{n,d}$?

$$\underbrace{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}}_{X_1} \times \underbrace{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}}_{X_2} \times \quad \cdots \quad \times \underbrace{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}}_{X_d}$$

Split into $\sqrt{d}$ blocks of $\sqrt{d}$ matrices each.

Compute $\mathrm{IMM}_{n,\sqrt{d}}$ for each block.

Recombine by computing $\mathrm{IMM}_{n,\sqrt{d}}$ again.

Reduction to depth 4

Depth-4 circuits for $\mathrm{IMM}_{n,d}$?

$$\underbrace{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}}_{X_1} \times \underbrace{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}}_{X_2} \times \quad \cdots \quad \times \underbrace{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}}_{X_d}$$

Split into $\sqrt{d}$ blocks of $\sqrt{d}$ matrices each.

Compute $\mathrm{IMM}_{n,\sqrt{d}}$ for each block.

Recombine by computing $\mathrm{IMM}_{n,\sqrt{d}}$ again.

$\mathrm{IMM}_{n,d}$ is computed by a $\sum\prod\sum\prod$ circuit of size $n^{O(\sqrt{d})}$.

Are algebraic constant depth circuits a weak model?

- They can't be simulated by constant-depth Boolean circuits
- $\sum\prod\sum$ can compute $\mathrm{ESym}_{n,d}$ in a non-homogeneous way
- Can simulate general Algebraic Circuits with a subexponential cost!

[VSBR – AV/K/T – GKKS]

If P can be computed by a circuit of size s,
then it can be computed by a

- log-depth circuit of size $\mathrm{poly}(sN)$,
- homogeneous $\sum\prod\sum\prod$ circuit of size $(sN)^{O(\sqrt{d})}$,

Are algebraic constant depth circuits a weak model?

- They can't be simulated by constant-depth Boolean circuits
- $\sum\prod\sum$ can compute $\mathrm{ESym}_{n,d}$ in a non-homogeneous way
- Can simulate general Algebraic Circuits with a subexponential cost!

[VSBR – AV/K/T – GKKS]

If P can be computed by a circuit of size s,
then it can be computed by a

- log-depth circuit of size $\mathrm{poly}(sN)$,
- homogeneous $\sum\prod\sum\prod$ circuit of size $(sN)^{O(\sqrt{d})}$,
- $\sum\prod\sum$ circuit of size $(sN)^{O(\sqrt{d})}$.

Constant Depth Lower Bounds

Boolean circuit lower bounds.

Strong lower bounds for constant-depth Boolean circuits known since the 80s.
[Ajtai 83, FSS 84, Håstad 86, Razborov 86, Smolensky 87].

Constant Depth Lower Bounds

Boolean circuit lower bounds.

Strong lower bounds for constant-depth Boolean circuits known since the 80s.
[Ajtai 83, FSS 84, Håstad 86, Razborov 86, Smolensky 87].

Algebraic circuit lower bounds.

The size of $\sum\prod$ formulas is just the number of monomials.
(Ex: $P(x_1, \ldots, x_N) = \sum_{S\subseteq[N]} \prod_{i\in S} x_i$ has 2^N monomials.)

Best known lower bound for $\sum\prod\sum$ circuits is $\Omega(N^3/\log^2 N)$
[Kayal,Saha,T.,2016].
Best known lower bound for $\sum\prod\sum\prod$ circuits is $\Omega(N^{2.5})$
[Gupta,Saha,Thankey,2020].
For depth $\Delta \geq 5$, lower bound in $N^{1+\Omega(1/\Delta)}$
[Shoup,Smolensky,96,Raz10].

Constant Depth Lower Bounds

Boolean circuit lower bounds.

Strong lower bounds for constant-depth Boolean circuits known since the 80s.
[Ajtai 83, FSS 84, Håstad 86, Razborov 86, Smolensky 87].

Algebraic circuit lower bounds.

Exp. lower bounds

The size of $\sum\prod$ formulas is just the number of monomials.
(Ex: $P(x_1, \ldots, x_N) = \sum_{S\subseteq[N]} \prod_{i\in S} x_i$ has 2^N monomials.)

No superpolynomial lower bounds

Best known lower bound for $\sum\prod\sum$ circuits is $\Omega(N^3/\log^2 N)$ [Kayal,Saha,T.,2016].
Best known lower bound for $\sum\prod\sum\prod$ circuits is $\Omega(N^{2.5})$ [Gupta,Saha,Thankey,2020].
For depth $\Delta \geq 5$, lower bound in $N^{1+\Omega(1/\Delta)}$ [Shoup,Smolensky,96,Raz10].

Superpolynomial Lower Bounds against Constant Depth Circuits

Main Theorem

Let n, d be growing parameters with $d \leq \log n$.
Assume $\mathbb{F}$ is of characteristic 0.

Superpolynomial Lower Bounds against Constant Depth Circuits

Main Theorem

Let n, d be growing parameters with $d \leq \log n$.
Assume $\mathbb{F}$ is of characteristic 0.

Any depth-Γ circuit for $\mathrm{IMM}_{n,d}$ must have size $n^{d^{\varepsilon_\Gamma}}$
where ε_Γ depends only on Γ.
Any depth-Γ circuit for Det_n must have size $n^{(\log n)^{\varepsilon_\Gamma}}$.

If $\Gamma = 3$, we have $\varepsilon_3 = 1/2$ (optimal for IMM).

If $\Gamma = 4$, we have $\varepsilon_4 = 1/4$.

Consequence: Polynomial Identity Testing

Subexponential time PIT

Given black-box access to a constant-depth poly(N)-size circuit computing a polynomial P,

Consequence: Polynomial Identity Testing

Subexponential time PIT

Given black-box access to a constant-depth poly(N)-size circuit computing a polynomial P,
there is a deterministic algorithm for checking whether $P \equiv 0$ that runs in subexponential time
(i.e., $N^{O(N^{\mu})}$ for any $\mu > 0$).

Consequence: Polynomial Identity Testing

Subexponential time PIT

Given black-box access to a constant-depth poly(N)-size circuit computing a polynomial P,
there is a deterministic algorithm for checking whether $P \equiv 0$ that runs in subexponential time
(i.e., $N^{O(N^{\mu})}$ for any $\mu > 0$).

Prior to this deterministic $n^{O(k)}$ time algorithm known for $\sum^{[k]} \prod \sum$ circuits. [Saxena,Seshadhri,2012]

Algebraic hardness vs. randomness (by [Chou,Kumar,Solomon,2018]) + our lower bound.

Builds on [Kabanets,Impagliazzo,2004], [Dvir,Shpilka,Yehudayoff,2009].

$$\begin{tikzcd}
\text{Lower bounds against general formulas} \\
\text{Lower bounds against weaker formulas} \arrow[u, "\text{Escalation}"']
\end{tikzcd}$$

Homogeneous/Set-multilinear restrictions

Set-multilinear polynomials

$P \in \mathbb{F}_{\mathsf{sm}}[X_1, \ldots, X_d]$, where $X_1, \ldots, X_d$ are sets of variables.

Each monomial uses exactly one variable per set.

Homogeneous/Set-multilinear restrictions

Set-multilinear polynomials

$P \in \mathbb{F}_{\mathrm{sm}}[X_1, \ldots, X_d]$, where $X_1, \ldots, X_d$ are sets of variables.

Each monomial uses exactly one variable per set.

Examples

$\mathrm{IMM}_{n,d}$

$$\underset{\mathrm{A}}{\begin{pmatrix} \boxed{} & & \\ & & \\ & & \end{pmatrix}} = \underset{X_1}{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}} \times \underset{X_2}{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}} \times \cdots \times \underset{X_d}{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}}$$

Homogeneous/Set-multilinear restrictions

Set-multilinear polynomials

$P \in \mathbb{F}_{\mathrm{sm}}[X_1, \ldots, X_d]$, where $X_1, \ldots, X_d$ are sets of variables.

Each monomial uses exactly one variable per set.

Examples

$\mathrm{IMM}_{n,d}$

$$\underset{\mathsf{A}}{\begin{pmatrix} \square & & \\ & & \\ & & \end{pmatrix}} = \underset{X_1}{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}} \times \underset{X_2}{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}} \times \cdots \times \underset{X_d}{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}}$$

$$\mathrm{PIP}_{n,d} = \langle X_1, X_2 \rangle \times \langle X_3, X_4 \rangle \times \ldots \times \langle X_{d-1}, X_d \rangle$$

Homogeneous/Set-multilinear restrictions

Set-multilinear polynomials

$P \in \mathbb{F}_{\mathsf{sm}}[X_1, \ldots, X_d]$, where $X_1, \ldots, X_d$ are sets of variables.

Each monomial uses exactly one variable per set.

Examples

$\mathrm{IMM}_{n,d}$

$$\underset{A}{\begin{pmatrix} \boxed{} & & \\ & & \\ & & \end{pmatrix}} = \underset{X_1}{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}} \times \underset{X_2}{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}} \times \cdots \times \underset{X_d}{\begin{pmatrix} & & \\ & & \\ & & \end{pmatrix}}$$

$\mathrm{PIP}_{n,d} = \langle X_1, X_2 \rangle \times \langle X_3, X_4 \rangle \times \ldots \times \langle X_{d-1}, X_d \rangle$

Homogeneous circuits/formulas

All gates compute homogeneous polynomials.

Set-multilinear circuits/formulas

All gates compute set-multilinear polynomials.

Homogeneization (Raz's approach)

Let $P(x_1, \ldots, x_N)$ be a set-multilinear polynomial of degree d.

[Raz 2009]

Formula of size s computing P

$$\Big\downarrow \text{Efficient conversion}$$

Set-multilinear formula computing P of size $\mathrm{poly}(s) \cdot (\log s)^{O(d)}$

Homogeneization (Raz's approach)

Let $P(x_1, \ldots, x_N)$ be a set-multilinear polynomial of degree $d = O(\log N / \log\log N)$.

[Raz 2009]

Formula of size s computing P

$$\Big\downarrow \text{ Efficient conversion}$$

Set-multilinear formula computing P of size $\mathrm{poly}(s) \cdot (\log s)^{O(d)}$

Homogeneization (Raz's approach)

Let $P(x_1, \ldots, x_N)$ be a set-multilinear polynomial of degree $d = O(\log N / \log\log N)$.

[Raz 2009]

Formula of size $\mathrm{poly}(N)$ computing P

$$\Big\downarrow \text{ Efficient conversion}$$

Set-multilinear formula computing P of size $\mathrm{poly}(N)$

Homogeneization (Raz's approach)

Let $P(x_1, \ldots, x_N)$ be a set-multilinear polynomial of degree $d = O(\log N / \log\log N)$.

[Raz 2009]

Any formula computing P needs superpolynomial size

Escalation

Set-multilinear formula computing P needs size $N^{\omega_d(1)}$

Homogeneization (Raz's approach)

Let $P(x_1, \ldots, x_N)$ be a set-multilinear polynomial of degree $d = O(\log N / \log\log N)$.

[Raz 2009]

Any formula computing P needs superpolynomial size

Caveat: Raz's transformation does not work for constant depth.

Escalation

Set-multilinear formula computing P needs size $N^{\omega_d(1)}$

Low degree regime - the blow-ups in size are all polynomial

Assumptions $d < \sqrt{\log n}$ and $\mathrm{char}(\mathbb{F}) \neq 0$.

Low degree regime - the blow-ups in size are all polynomial

Assumptions $d < \sqrt{\log n}$ and $\mathrm{char}(\mathbb{F}) \neq 0$.

Parallelization

- Parallelization of the circuits to depth $O(\log d)$. [VSBR83]
- Parallelization of the formulas to depth $O(\log s)$. [BKM73]
- Parallelization of the homogeneous formulas to depth $O(\log d)$. [FLMST23]

Low degree regime - the blow-ups in size are all polynomial

Assumptions $d < \sqrt{\log n}$ and $\mathrm{char}(\mathbb{F}) \neq 0$.

Parallelization

Parallelization of the circuits to depth $O(\log d)$. [VSBR83]

Parallelization of the formulas to depth $O(\log s)$. [BKM73]

Parallelization of the homogeneous formulas to depth $O(\log d)$. [FLMST23]

Structural results

Homogeneization/Set-multilinearization of the circuits. [Str73,NW97]

Idem for formulas. [Raz13]

Hom./S-multilinearization of the circuits
where the depth is multiplied by at most 2.
[SW01,CKSV16,LST21]

Low degree regime - the blow-ups in size are all polynomial

Assumptions $d < \sqrt{\log n}$ and $\mathrm{char}(\mathbb{F}) \neq 0$.

Parallelization

- Parallelization of the circuits to depth $O(\log d)$. [VSBR83]
- Parallelization of the formulas to depth $O(\log s)$. [BKM73]
- Parallelization of the homogeneous formulas to depth $O(\log d)$. [FLMST23]

Structural results

- Homogeneization/Set-multilinearization of the circuits. [Str73,NW97]
- Idem for formulas. [Raz13]
- Hom./S-multilinearization of the circuits where the depth is multiplied by at most 2. [SW01,CKSV16,LST21]

Sufficient to prove $n^{\omega(d)}$ lower bounds for set-multilinear formulas of depth $O(\log d)$!

Non-FPT Lower Bounds

Known lower bounds

Known set-multiliear formula lower bounds for constant depth. [NW 95, Raz 2009, RY 2009]

Non-FPT Lower Bounds

Known lower bounds

Known set-multiliear formula lower bounds for constant depth. [NW 95, Raz 2009, RY 2009]

$$\exp(\Omega(d)) \cdot \mathrm{poly}(N)$$

Non-FPT Lower Bounds

Known lower bounds

Known set-multiliear formula lower bounds for constant depth. [NW 95, Raz 2009, RY 2009]

$$\exp(\Omega(d)) \cdot \mathrm{poly}(N)$$

For escalation to work, we need:

Non-FPT Lower Bounds

Known lower bounds

Known set-multiliear formula lower bounds for constant depth. [NW 95, Raz 2009, RY 2009]

$$\exp(\Omega(d)) \cdot \mathrm{poly}(N)$$

For escalation to work, we need:

$$N^{\Omega(f(d))}$$

Our Lower Bound

A non-FPT lower bound for set-multilinear formulas.

Set-multilinear formula lower bound

Let $d \leq O(\log n)$.
For any $\Delta \geq 1$ any set-multilinear formula C computing $\mathrm{IMM}_{n,d}$ of depth Δ must have size $n^{d^{\varepsilon^{\Delta}}}$.

First case $\Delta = 5$: bound in $n^{\Omega(\sqrt{d})}$

Case $\Gamma = 3$

We just stated:

Set-multilinear formula lower bound

Let $d \leq O(\log n)$. Any set-multilinear formula C computing $\mathsf{IMM}_{n,d}$ of depth 5 must have size $n^{\Omega(\sqrt{d})}$.

Case $\Gamma = 3$

We just stated:

Set-multilinear formula lower bound

Let $d \leq O(\log n)$. Any set-multilinear formula C computing $\mathrm{IMM}_{n,d}$ of depth 5 must have size $n^{\Omega(\sqrt{d})}$.

In particular,

General formula lower bound

Let n, d be growing parameters with $d = o(\log n)$.
Assume $\mathbb{F}$ is characteristic 0.
Any algebraic circuits of depth 3 computing $\mathrm{IMM}_{n,d}$ must have size $n^{\Omega(\sqrt{d})}$.

Techniques

A typical lower bound proof

The lower bound proof outline.

- Come up with a measure $\mu : \mathbb{F}_{\mathsf{sm}}[X_1, \dots, X_d] \to \mathbb{R}_{\geq 0}$.
- Show that $\mu(\mathsf{IMM}_{n,d})$ is large.
- Show that $\mu(\text{sm. } \sum\prod\sum\prod\sum)$ is small.

Partial Derivative Measure

Nisan and Wigderson [NW 95]

Partial Derivative Measure

Nisan and Wigderson [NW 95]

Partition $[d]$ into $\mathcal{P}$ and $\mathcal{N}$.

$M^{\mathcal{P}}$ multilinear monomials over $(X_i : i \in \mathcal{P})$.

$M^{\mathcal{N}}$ multilinear monomials over $(X_i : i \in \mathcal{N})$.

Partial Derivative Measure

Nisan and Wigderson [NW 95]

Partition $[d]$ into $\mathcal{P}$ and $\mathcal{N}$.

$M^{\mathcal{P}}$ multilinear monomials over $(X_i : i \in \mathcal{P})$.

$M^{\mathcal{N}}$ multilinear monomials over $(X_i : i \in \mathcal{N})$.

For a polynomial f, define matrix M_f as follows.

$$\begin{array}{ccc} & M^{\mathcal{N}} \longrightarrow & \\ M^{\mathcal{P}} & m_2 & \\ \downarrow \quad m_1 & \begin{pmatrix} & & \\ & \boxed{} & \\ & & \end{pmatrix} & \end{array}$$

Coeff of $m_1 \cdot m_2$ in f

Partial Derivative Measure

Nisan and Wigderson [NW 95]

Partition $[d]$ into $\mathcal{P}$ and $\mathcal{N}$.

$M^{\mathcal{P}}$ multilinear monomials over $(X_i : i \in \mathcal{P})$.

$M^{\mathcal{N}}$ multilinear monomials over $(X_i : i \in \mathcal{N})$.

For a polynomial f, define matrix M_f as follows.

$$\begin{array}{ccc} & M^{\mathcal{N}} \longrightarrow & \\ M^{\mathcal{P}} & & m_2 \\ \downarrow & & \\ & m_1 & \left(\begin{array}{ccc} & & \\ & & \\ & \boxed{} & \\ & & \end{array}\right) \end{array}$$

Coeff of $m_1 \cdot m_2$ in f

The Partial Derivative Measure is the $\mathrm{rank}(M_f)$.

Properties of μ

$\mu : \mathbb{F}_{\mathsf{sm}}[X_1, \ldots, X_d] \to \mathbb{N}$

μ is sub-additive: $\mu(f+g) \leq \mu(f) + \mu(g)$

μ is multiplicative: $\mu(fg) = \mu(f)\mu(g)$

$\mu(f) \leq \min(M^{\mathcal{P}}, M^{\mathcal{N}})$

A typical lower bound proof

The lower bound proof outline.

- Come up with a measure $\mu : \mathbb{F}_{\mathsf{sm}}[X_1, \ldots, X_d] \to \mathbb{R}_{\geq 0}$. ✔
- Show that $\mu(\mathsf{IMM}_{n,d})$ is large.
- Show that $\mu(\text{sm. } \sum\prod\sum\prod\sum)$ is small.

The measure applied to $\mathrm{IMM}_{n,d}$

Recall that

$$\mathrm{IMM}_{n,d} = \sum_{i_1,\ldots,i_{d-1}\in[n]} X^{(1)}_{1,i_1}\cdot X^{(2)}_{i_1,i_2}\cdot X^{(3)}_{i_2,i_3}\cdots X^{(d)}_{i_{d-1},1}.$$

For $\mathcal{P} = \{i \mid i \text{ odd}\}$ and $\mathcal{N} = \{j \mid j \text{ even}\}$ (Assume d even)

The measure applied to $\mathrm{IMM}_{n,d}$

Recall that

$$\mathrm{IMM}_{n,d} = \sum_{i_1,\ldots,i_{d-1}\in[n]} X^{(1)}_{1,i_1} \cdot X^{(2)}_{i_1,i_2} \cdot X^{(3)}_{i_2,i_3} \cdots X^{(d)}_{i_{d-1},1}.$$

For $\mathcal{P} = \{i \mid i \text{ odd}\}$ and $\mathcal{N} = \{j \mid j \text{ even}\}$ (Assume d even)

$$M^{\mathcal{P}} \downarrow \qquad M^{\mathcal{N}} \longrightarrow$$

$$\begin{array}{cc} & m_2 \\ m_1 & \begin{pmatrix} & & \\ & \boxed{} & \\ & & \end{pmatrix} \end{array}$$

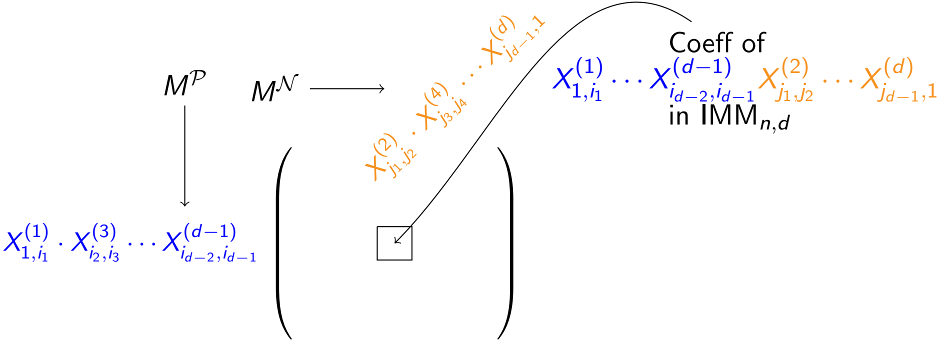
Coeff of $m_1 \cdot m_2$ in $\mathrm{IMM}_{n,d}$

The measure applied to $\mathrm{IMM}_{n,d}$

Recall that

$$\mathrm{IMM}_{n,d} = \sum_{i_1,\ldots,i_{d-1}\in[n]} X^{(1)}_{1,i_1}\cdot X^{(2)}_{i_1,i_2}\cdot X^{(3)}_{i_2,i_3}\cdots X^{(d)}_{i_{d-1},1}.$$

For $\mathcal{P} = \{i \mid i \text{ odd}\}$ and $\mathcal{N} = \{j \mid j \text{ even}\}$ (Assume d even)



The measure applied to $\mathrm{IMM}_{n,d}$

Recall that

$$\mathrm{IMM}_{n,d} = \sum_{i_1,\ldots,i_{d-1}\in[n]} X^{(1)}_{1,i_1} \cdot X^{(2)}_{i_1,i_2} \cdot X^{(3)}_{i_2,i_3} \cdots X^{(d)}_{i_{d-1},1}.$$

For $\mathcal{P} = \{i \mid i \text{ odd}\}$ and $\mathcal{N} = \{j \mid j \text{ even}\}$ (Assume d even)

$M^{\mathcal{P}}$ $\quad M^{\mathcal{N}} \longrightarrow$ $X^{(2)}_{j_1,j_2} \cdot X^{(4)}_{j_3,j_4} \cdots X^{(d)}_{j_{d-1},1}$

Coeff of $X^{(1)}_{1,i_1} \cdots X^{(d-1)}_{i_{d-2},i_{d-1}} X^{(2)}_{j_1,j_2} \cdots X^{(d)}_{j_{d-1},1}$ in $\mathrm{IMM}_{n,d}$

$$X^{(1)}_{1,i_1} \cdot X^{(3)}_{i_2,i_3} \cdots X^{(d-1)}_{i_{d-2},i_{d-1}} \quad \begin{pmatrix} & & \\ & \boxed{} & \\ & & \end{pmatrix} \quad \begin{cases} = 1 \text{ if } (i_1,..,i_{d-1}) = (j_1,..,j_{d-1} \\ = 0 \text{ otherwise.} \end{cases}$$

The measure applied to $\mathrm{IMM}_{n,d}$

Recall that

$$\mathrm{IMM}_{n,d} = \sum_{i_1,\ldots,i_{d-1}\in[n]} X^{(1)}_{1,i_1}\cdot X^{(2)}_{i_1,i_2}\cdot X^{(3)}_{i_2,i_3}\cdots X^{(d)}_{i_{d-1},1}.$$

For $\mathcal{P} = \{i \mid i \text{ odd}\}$ and $\mathcal{N} = \{j \mid j \text{ even}\}$ (Assume d even)

$$M^{\mathcal{P}} \downarrow \qquad M^{\mathcal{N}} \longrightarrow$$

$$\begin{array}{cc} & X^{(2)}_{i_1,i_2}\cdot X^{(4)}_{i_3,i_4}\cdots X^{(d)}_{i_{d-1},1} \\ X^{(1)}_{1,i_1}\cdot X^{(3)}_{i_2,i_3}\cdots X^{(d-1)}_{i_{d-2},i_{d-1}} & \begin{pmatrix} 1 & & \\ & \boxed{1} & \\ & & 1 \end{pmatrix} \end{array} \begin{cases} = 1 \text{ if } (i_1,..,i_{d-1}) = (j_1,..,j_{d-1}) \\ = 0 \text{ otherwise.} \end{cases}$$

Permutation matrix

The measure applied to $\mathrm{IMM}_{n,d}$

Recall that

$$\mathrm{IMM}_{n,d} = \sum_{i_1,\ldots,i_{d-1}\in[n]} X^{(1)}_{1,i_1}\cdot X^{(2)}_{i_1,i_2}\cdot X^{(3)}_{i_2,i_3}\cdots X^{(d)}_{i_{d-1},1}.$$

For $\mathcal{P} = \{i \mid i \text{ odd}\}$ and $\mathcal{N} = \{j \mid j \text{ even}\}$ (Assume d even)

$$M^{\mathcal{P}} \big\downarrow \qquad M^{\mathcal{N}} \longrightarrow$$

$$\begin{array}{cc} & X^{(2)}_{i_1,i_2}\cdot X^{(4)}_{i_3,i_4}\cdots X^{(d)}_{i_{d-1},1} \\ X^{(1)}_{1,i_1}\cdot X^{(3)}_{i_2,i_3}\cdots X^{(d-1)}_{i_{d-2},i_{d-1}} & \begin{pmatrix} 1 & & \\ & \boxed{1} & \\ & & 1 \end{pmatrix} \end{array} \qquad \begin{cases} = 1 \text{ if } (i_1,..,i_{d-1}) = (j_1,..,j_{d-1}) \\ = 0 \text{ otherwise.} \end{cases}$$

Permutation matrix

The matrix is full-rank! $\mathrm{rk}(\mathrm{IMM}_{n,d}) = n^{d-1}$.

A typical lower bound proof

The lower bound proof outline.

- Come up with a measure $\mu : \mathbb{F}_{\mathsf{sm}}[X_1, \ldots, X_d] \to \mathbb{R}_{\geq 0}$. ✔
- Show that $\mu(\mathsf{IMM}_{n,d})$ is large. ✔
- Show that $\mu(\text{sm. } \sum\prod\sum\prod\sum)$ is small.

$\sum\prod\sum$ set-multilinear formulas

Let $(X_1, \ldots, X_d)$ be a partition of variables.

$$F(X) = \sum_{i=1}^{s} \prod_{j=1}^{d} \ell_{i,j}(X_j)$$

each $\ell_{i,j}$ homogeneous linear polynomial over X_j.

$\sum\prod\sum$ set-multilinear formulas

Let $(X_1, \ldots, X_d)$ be a partition of variables.

$$F(X) = \sum_{i=1}^{s} \prod_{j=1}^{d} \ell_{i,j}(X_j)$$

each $\ell_{i,j}$ homogeneous linear polynomial over X_j.

For each $i \in [s], j \in [d]$, $\mu(\ell_{i,j}(X_j))$ at most 1.

$\sum\prod\sum$ set-multilinear formulas

Let $(X_1, \ldots, X_d)$ be a partition of variables.

$$F(X) = \sum_{i=1}^{s} \prod_{j=1}^{d} \ell_{i,j}(X_j)$$

each $\ell_{i,j}$ homogeneous linear polynomial over X_j.

For each $i \in [s], j \in [d]$, $\mu\left(\ell_{i,j}(X_j)\right)$ at most 1.

For each $i \in [s]$, $\mu\left(\prod_{j=1}^{d} \ell_{i,j}(X_j)\right)$ at most 1.

$\sum\prod\sum$ set-multilinear formulas

Let $(X_1, \ldots, X_d)$ be a partition of variables.

$$F(X) = \sum_{i=1}^{s} \prod_{j=1}^{d} \ell_{i,j}(X_j)$$

each $\ell_{i,j}$ homogeneous linear polynomial over X_j.

For each $i \in [s], j \in [d]$, $\mu\left(\ell_{i,j}(X_j)\right)$ at most 1.

For each $i \in [s]$, $\mu\left(\prod_{j=1}^{d} \ell_{i,j}(X_j)\right)$ at most 1.

By subadditivity of rank, $\mu(F(X))$ at most s.

$\sum\prod\sum$ set-multilinear formulas

Let $(X_1, \ldots, X_d)$ be a partition of variables.

$$F(X) = \sum_{i=1}^{s} \prod_{j=1}^{d} \ell_{i,j}(X_j)$$

each $\ell_{i,j}$ homogeneous linear polynomial over X_j.

For each $i \in [s], j \in [d]$, $\mu\left(\ell_{i,j}(X_j)\right)$ at most 1.

For each $i \in [s]$, $\mu\left(\prod_{j=1}^{d} \ell_{i,j}(X_j)\right)$ at most 1.

By subadditivity of rank, $\mu(F(X))$ at most s.

Conclusion: $\sum\prod\sum$ s.m. form. for $\mathrm{IMM}_{n,d}$ has size $\geq n^{d-1}$.

$\sum\prod\sum\prod$ set-multilinear formulas

Product of Inner Products Polynomial.

Let $X_j = \{x_{j,1}, \ldots, x_{j,m}\}$ for $j \in [d]$.

$$\mathrm{PIP}(X_1, \ldots, X_d) = \prod_{j=1}^{d/2} \left(\sum_{k=1}^{m} x_{2j-1,k} \cdot x_{2j,k} \right)$$

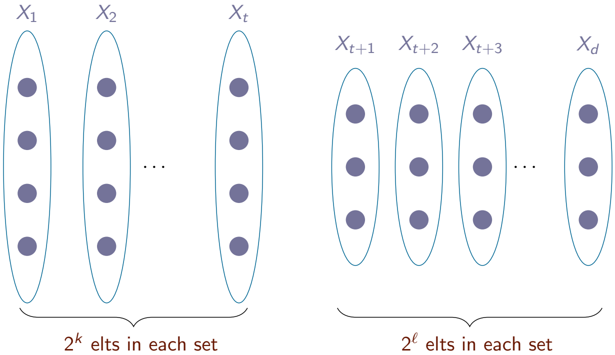
PIP has product-depth 2 set-multilinear formula of size $O(md)$.

For $\mathcal{P} = \{i \mid i \text{ odd}\}$ and $\mathcal{N} = \{i \mid i \text{ even}\}$,

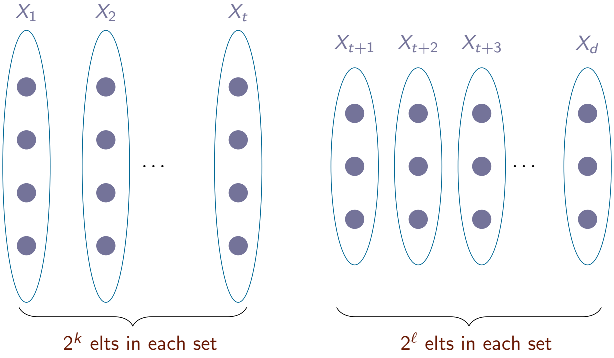
M_{PIP} is a permutation matrix.

rk(PIP) is full.

Idea: Different set sizes

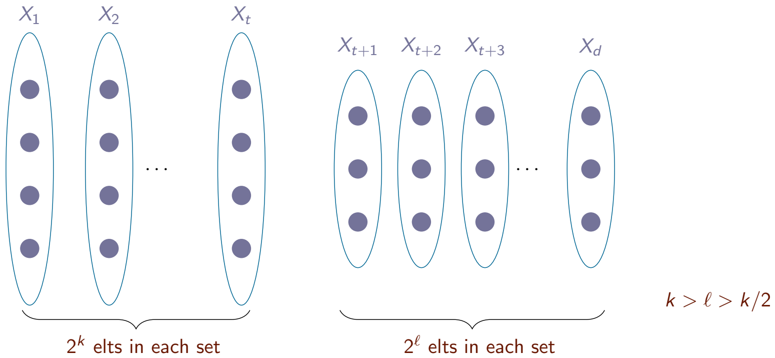


Idea: Different set sizes



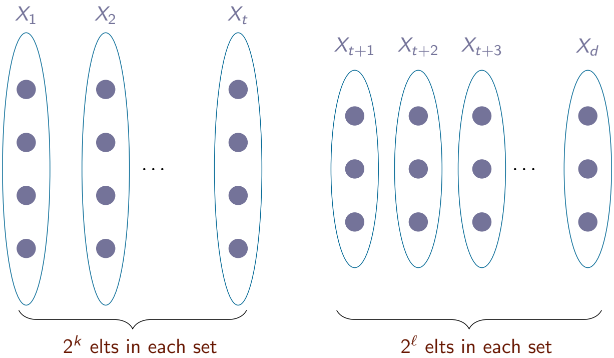
$k > \ell > k/2$

Idea: Different set sizes



We want to ensure $|M^{\mathcal{P}}| = |M^{\mathcal{N}}|$.

Idea: Different set sizes



$k > \ell > k/2$
$kt = (d-t)\ell$

We want to ensure $|M^{\mathcal{P}}| = |M^{\mathcal{N}}|$.

$\sum\prod\sum\prod\sum$ set-multilinear formulas

- Sets of size 2^k, 2^ℓ
- $k > \ell > k/2$
- Full rank $= 2^{kt}$

$$\sum \quad \overset{F}{\boxed{\prod_j \underbrace{\left(\sum\prod\sum\right)}_{F_j}}}$$

$\sum\prod\sum\prod\sum$ set-multilinear formulas

- Sets of size 2^k, 2^ℓ
- $k > \ell > k/2$
- Full rank $= 2^{kt}$

$$\overset{F}{\sum \quad \boxed{\prod_j \underbrace{\left(\sum\prod\sum\right)}_{F_j}}}$$

Focus on one term F, which is $F_1 \times F_2 \times \ldots \times F_r$.

$\sum\prod\sum\prod\sum$ set-multilinear formulas

- Sets of size 2^k, 2^ℓ
- $k > \ell > k/2$
- Full rank $= 2^{kt}$

$$\overbrace{\sum \boxed{\prod_j \underbrace{\left(\sum\prod\sum\right)}_{F_j}}}^{F}$$

Focus on one term F, which is $F_1 \times F_2 \times \ldots \times F_r$.

Sufficient to show $\mu(F) \leq \frac{2^{kt}}{n^{\sqrt{d}/100}} = \frac{\sqrt{2^{kt}2^{\ell(d-t)}}}{2^{k\sqrt{d}/100}}$.

Each F_j is a $(\sum\prod\sum)$ set-multilinear formula.

It covers p_j $\mathcal{P}$-variables-sets and q_j from $\mathcal{N}$.

$\sum\prod\sum\prod\sum$ set-multilinear formulas

- Sets of size 2^k, 2^ℓ
- $k > \ell > k/2$
- Full rank $= 2^{kt}$

$$\overset{F}{\boxed{\prod_j \underbrace{\left(\sum\prod\sum\right)}_{F_j}}}$$

$$\sum \boxed{\prod_j \underbrace{\left(\sum\prod\sum\right)}_{F_j}}$$

Focus on one term F, which is $F_1 \times F_2 \times \ldots \times F_r$.

Sufficient to show $\mu(F) \leq \frac{2^{kt}}{n^{\sqrt{d}/100}} = \frac{\sqrt{2^{kt}2^{\ell(d-t)}}}{2^{k\sqrt{d}/100}}$.

Each F_j is a $(\sum\prod\sum)$ set-multilinear formula.

It covers p_j $\mathcal{P}$-variables-sets and q_j from $\mathcal{N}$.

So $\mu(F) = \quad \mu(F_1) \quad \times \cdots \times \quad \mu(F_r)$

$$= \frac{\sqrt{2^{kp_1}2^{\ell q_1}}}{\mathrm{Loss}(F_1)} \times \cdots \times \frac{\sqrt{2^{kp_r}2^{\ell q_r}}}{\mathrm{Loss}(F_r)}$$

$\sum\prod\sum\prod\sum$ set-multilinear formulas

$$\overbrace{\sum \boxed{\prod_j \underbrace{(\sum\prod\sum)}_{F_j}}}^{F}$$

- Sets of size 2^k, 2^ℓ
- $k > \ell > k/2$
- Full rank $= 2^{kt}$
- $\mu(F_j) = \frac{\sqrt{2^{kp_j}2^{\ell q_j}}}{\mathrm{Loss}(F_j)}$
- We want:

$$2^{k\sqrt{d}/100} \leq \prod \mathrm{Loss}(F_j)$$

Focus on one term F, which is $F_1 \times F_2 \times \ldots \times F_r$.

Sufficient to show $\mu(F) \leq \frac{2^{kt}}{n^{\sqrt{d}/100}} = \frac{\sqrt{2^{kt}2^{\ell(d-t)}}}{2^{k\sqrt{d}/100}}$.

Each F_j is a $(\sum\prod\sum)$ set-multilinear formula.

It covers p_j $\mathcal{P}$-variables-sets and q_j from $\mathcal{N}$.

So $\mu(F) = \mu(F_1) \times \cdots \times \mu(F_r)$

$$= \frac{\sqrt{2^{kp_1}2^{\ell q_1}}}{\mathrm{Loss}(F_1)} \times \cdots \times \frac{\sqrt{2^{kp_r}2^{\ell q_r}}}{\mathrm{Loss}(F_r)}$$

$\sum\prod\sum\prod\sum$ set-multilinear formulas

$$F$$

$$\underbrace{\boxed{\prod_j \left(\sum\prod\sum\right)}}_{F_j}$$

- Sets of size 2^k, 2^ℓ
- $k > \ell > k/2$
- Full rank $= 2^{kt}$
- $\mu(F_j) = \frac{\sqrt{2^{kp_j}2^{\ell q_j}}}{\mathrm{Loss}(F_j)}$
- We want:

$$2^{k\sqrt{d}/100} \leq \prod \mathrm{Loss}(F_j)$$

$\sum\prod\sum\prod\sum$ set-multilinear formulas

$$F$$

$$\boxed{\textstyle\prod_j \underbrace{\left(\sum\prod\sum\right)}_{F_j}}$$

- Sets of size 2^k, 2^ℓ
- $k > \ell > k/2$
- Full rank $= 2^{kt}$
- $\mu(F_j) = \frac{\sqrt{2^{kp_j}2^{\ell q_j}}}{\mathrm{Loss}(F_j)}$
- We want:

$$2^{k\sqrt{d}/100} \leq \prod \mathrm{Loss}(F_j)$$

Case 1 There is an F_j with degree $\geq \sqrt{d}/2$.

$\sum\prod\sum\prod\sum$ set-multilinear formulas

- Sets of size 2^k, 2^ℓ
- $k > \ell > k/2$
- Full rank $= 2^{kt}$
- $\mu(F_j) = \frac{\sqrt{2^{kp_j}2^{\ell q_j}}}{\mathrm{Loss}(F_j)}$
- We want:

$$2^{k\sqrt{d}/100} \leq \prod \mathrm{Loss}(F_j)$$

$$F$$

$$\boxed{\prod_j \underbrace{\left(\sum\prod\sum\right)}_{F_j}}$$

Case 1 There is an F_j with degree $\geq \sqrt{d}/2$.

We saw $\mu(\sum\prod\sum) \leq \mathrm{size}(\sum\prod\sum)$.

If the size is $\geq 2^{k\sqrt{d}/50}$ 

Otherwise

$$2^{k\sqrt{d}/50} \geq \mu(F_j) = \frac{\sqrt{2^{kp_j}2^{\ell q_j}}}{\mathrm{Loss}(F_j)} \geq \frac{2^{k\sqrt{d}/8}}{\mathrm{Loss}(F_j)}.$$

$\sum\prod\sum\prod\sum$ set-multilinear formulas

F

$$\boxed{\prod_j \underbrace{\left(\sum\prod\sum\right)}_{F_j}}$$

- Sets of size 2^k, 2^ℓ
- $k > \ell > k/2$
- Full rank $= 2^{kt}$
- $\mu(F_j) = \frac{\sqrt{2^{kp_j}2^{\ell q_j}}}{\mathrm{Loss}(F_j)}$
- We want:

$$2^{k\sqrt{d}/100} \leq \prod \mathrm{Loss}(F_j)$$

Case 1 There is an F_j with degree $\geq \sqrt{d}/2$.

We saw $\mu(\sum\prod\sum) \leq \mathrm{size}(\sum\prod\sum)$.

If the size is $\geq 2^{k\sqrt{d}/50}$

Otherwise

$$2^{k\sqrt{d}/50} \geq \mu(F_j) = \frac{\sqrt{2^{kp_j}2^{\ell q_j}}}{\mathrm{Loss}(F_j)} \geq \frac{2^{k\sqrt{d}/8}}{\mathrm{Loss}(F_j)}.$$

Conclusion: $\mathrm{Loss}(F_j) \geq 2^{k\sqrt{d}/100}$.

$\sum\prod\sum\prod\sum$ set-multilinear formulas

$$F$$

$$\boxed{\textstyle\prod_j \underbrace{(\sum\prod\sum)}_{F_j}}$$

- Sets of size 2^k, 2^ℓ
- $k > \ell > k/2$
- Full rank $= 2^{kt}$
- $\mu(F_j) = \frac{\sqrt{2^{kp_j}2^{\ell q_j}}}{\mathrm{Loss}(F_j)}$
- We want:

$$2^{k\sqrt{d}/100} \le \prod \mathrm{Loss}(F_j)$$

Case 2 All F_j have degree $< \sqrt{d}/2$.

$\sum\prod\sum\prod\sum$ set-multilinear formulas

$$F$$

$$\underbrace{\boxed{\prod_j \left(\sum\prod\sum\right)}}_{F_j}$$

- Sets of size 2^k, 2^ℓ
- $k > \ell > k/2$
- Full rank $= 2^{kt}$
- $\mu(F_j) = \frac{\sqrt{2^{kp_j} 2^{\ell q_j}}}{\mathrm{Loss}(F_j)}$
- We want:

$$2^{k\sqrt{d}/100} \leq \prod \mathrm{Loss}(F_j)$$

Case 2 All F_j have degree $< \sqrt{d}/2$.

Let us choose $\ell = \lfloor k - k/(10\sqrt{d}) \rfloor$.

Focus on the ratio between the # of rows and of columns:

$$|kp_j - \ell q_j| > \frac{q_j k}{10\sqrt{d}}.$$

So $\mathrm{Loss}(F_j) \geq 2^{q_j k/(20\sqrt{d})}$.

Conclusion: $\prod \mathrm{Loss}(F_j) \geq \prod 2^{q_j k/(20\sqrt{d})} \geq 2^{k\sqrt{d}/40}$.

A typical lower bound proof

The lower bound proof outline.

- Come up with a measure $\mu : \mathbb{F}_{\mathsf{sm}}[X_1, \ldots, X_d] \to \mathbb{R}_{\geq 0}$. ✔
- Show that $\mu(\mathsf{IMM}_{n,d})$ is large. ✔
- Show that $\mu(\text{sm. } \sum\prod\sum\prod\sum)$ is small. ✔

We just showed:

Set-multilinear formula lower bound

Let $d \leq O(\log n)$. Any set-multilinear formula C computing $\mathrm{IMM}_{n,d}$ of depth 5 must have size $n^{\Omega(\sqrt{d})}$.

We just showed:

Set-multilinear formula lower bound

Let $d \leq O(\log n)$. Any set-multilinear formula C computing $\mathrm{IMM}_{n,d}$ of depth 5 must have size $n^{\Omega(\sqrt{d})}$.

In particular,

General formula lower bound

Let n, d be growing parameters with $d = o(\log n)$.
Assume $\mathbb{F}$ is characteristic 0.
Any algebraic circuits of depth 3 computing $\mathrm{IMM}_{n,d}$ must have size $n^{\Omega(\sqrt{d})}$.

General case

Set-multilinear formula lower bound

Let $d \leq O(\log n)$. Any set-multilinear formula C computing $\mathrm{IMM}_{n,d}$ of depth Δ must have size $n^{d^{\exp(-O(\Delta))}}$.

General case

Set-multilinear formula lower bound

Let $d \leq O(\log n)$. Any set-multilinear formula C computing $\mathrm{IMM}_{n,d}$ of depth Δ must have size $n^{d^{\exp(-O(\Delta))}}$.

In particular,

General formula lower bound

Let n, d be growing parameters with $d = o(\log n)$.
Assume $\mathbb{F}$ is characteristic 0.
Any algebraic circuits of depth Γ computing $\mathrm{IMM}_{n,d}$ must have size $n^{d^{\exp(-O(\Gamma))}}$.

Open Questions

Can the lower bound be improved? What about $n^{\Omega(d^{1/\Delta})}$?

Can we remove the characteristic 0 condition?

Can we get better lower bounds if we consider non-commutative computations?

Can combining known measures give better lower bounds?