Finding the Order of an ROABP

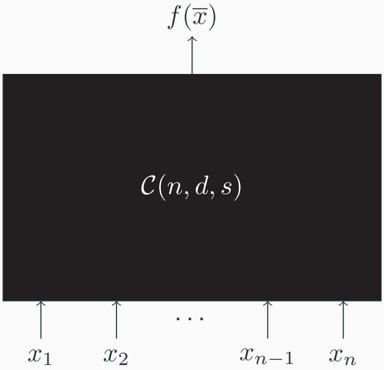
Can we proper-learn ROABPs?

Anamay Tengse

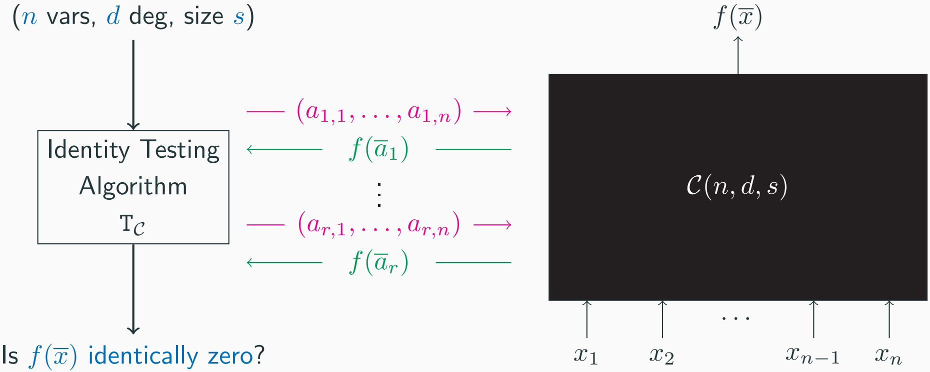
TIFR, June 11, 2025

joint work with Vishwas Bhargava, Pranjal Dutta and Sumanta Ghosh

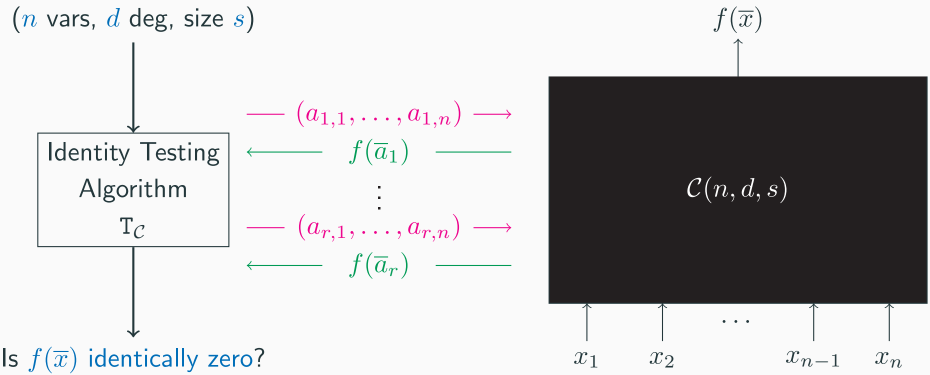
Polynomial Identity Testing (PIT)



Polynomial Identity Testing (PIT)



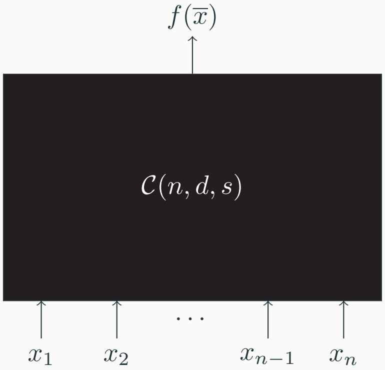
Polynomial Identity Testing (PIT)



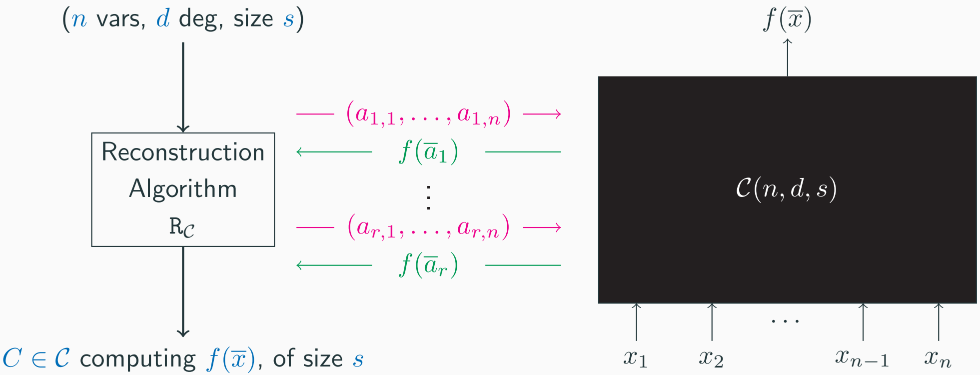
Known. Randomized, $\text{poly}(n, d, s)$-time algorithm for circuits.

Open. Deterministic algorithm for circuits, making fewer than d^n queries.

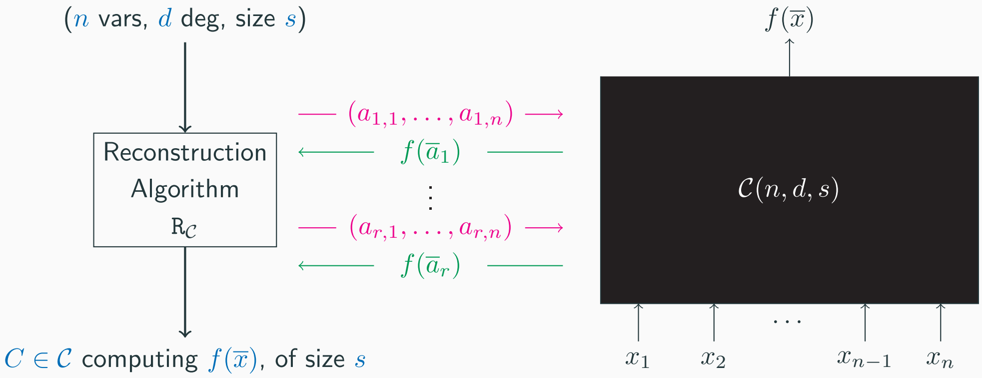
Reconstruction ("Proper-Learning") of Polynomials



Reconstruction (“Proper-Learning”) of Polynomials



Reconstruction ("Proper-Learning") of Polynomials

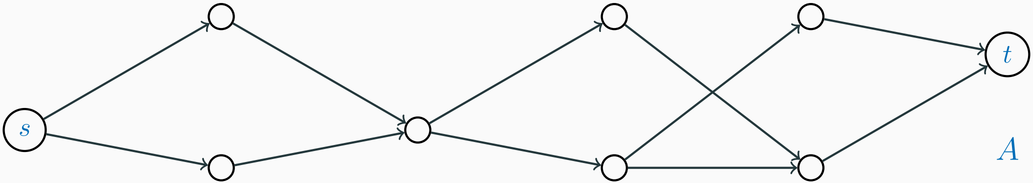


$\mathrm{R}_{\mathcal{C}}$ exists: "The class $\mathcal{C}$ can be proper-learnt in `<complexity-of-R>`"

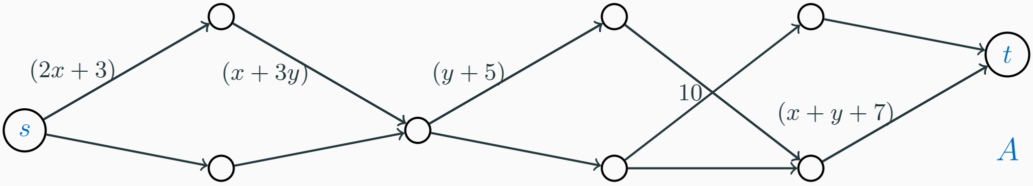
The Question

Read-once Oblivious ABPs

Algebraic Branching Programs (ABPs)

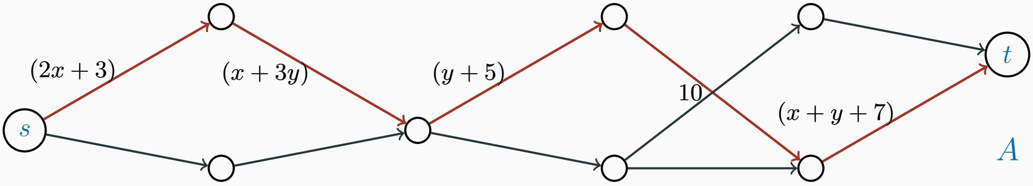


Algebraic Branching Programs (ABPs)



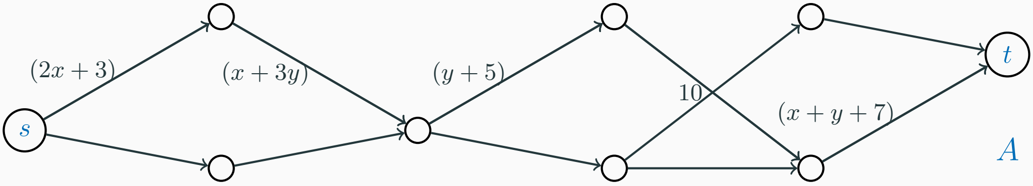
- Label on each edge: An linear polynomial in $\{x_1, x_2, \ldots, x_n\}$

Algebraic Branching Programs (ABPs)



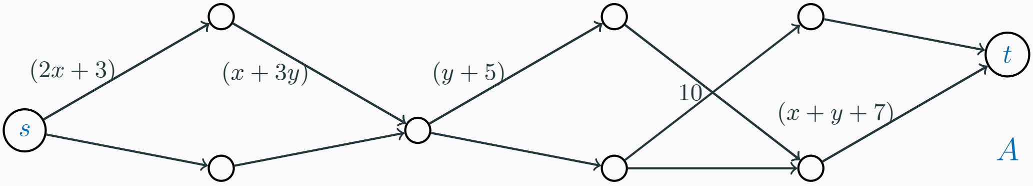
- Label on each edge: An linear polynomial in $\{x_1, x_2, \ldots, x_n\}$
- Polynomial computed by the path $p = \mathrm{wt}(p)$: Product of the edge labels on p

Algebraic Branching Programs (ABPs)



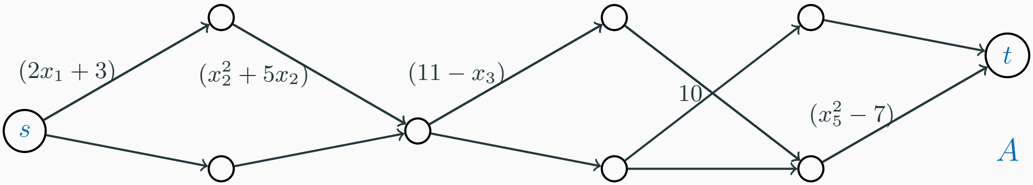
- Label on each edge: An linear polynomial in $\{x_1, x_2, \ldots, x_n\}$
- Polynomial computed by the path $p = \mathrm{wt}(p)$: Product of the edge labels on p
- Polynomial computed by the ABP: $f_A(\overline{x}) = \sum_p \mathrm{wt}(p)$ (sum over s to t paths)

Algebraic Branching Programs (ABPs)

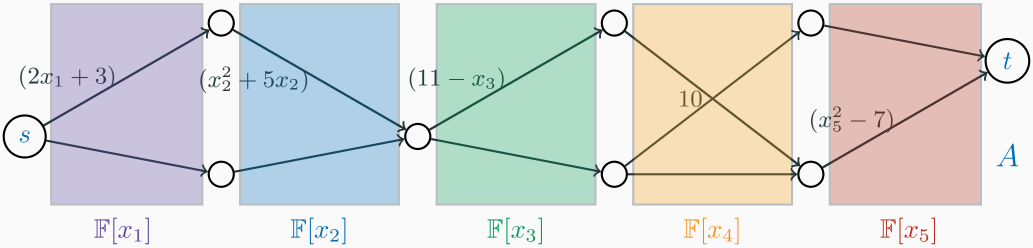


- Label on each edge: An linear polynomial in $\{x_1, x_2, \ldots, x_n\}$
- Polynomial computed by the path $p = \mathrm{wt}(p)$: Product of the edge labels on p
- Polynomial computed by the ABP: $f_A(\overline{x}) = \sum_p \mathrm{wt}(p)$ (sum over s to t paths)
- Size of the ABP: total number of vertices (9 in the example)

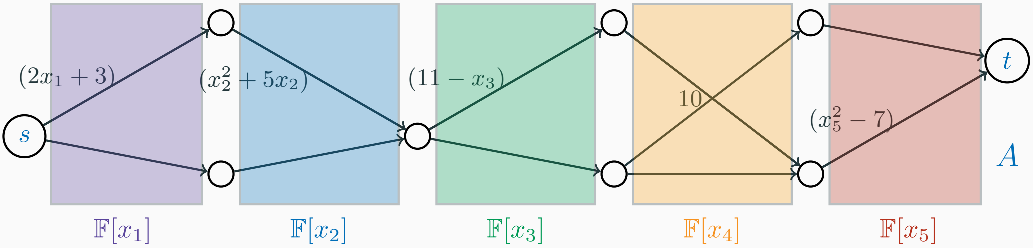
Read-once Oblivious ABPs



Read-once Oblivious ABPs



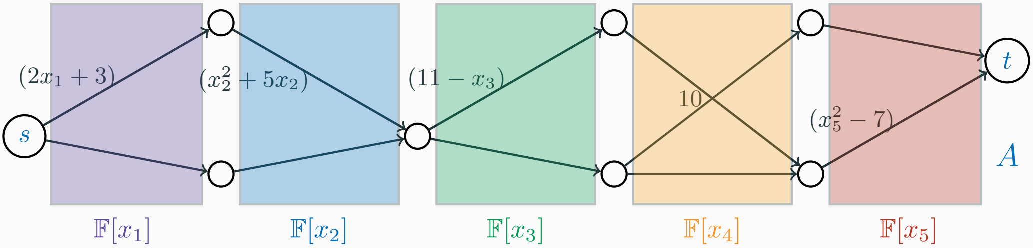
Read-once Oblivious ABPs



ROABP (n-variate, degree-d, width-w)

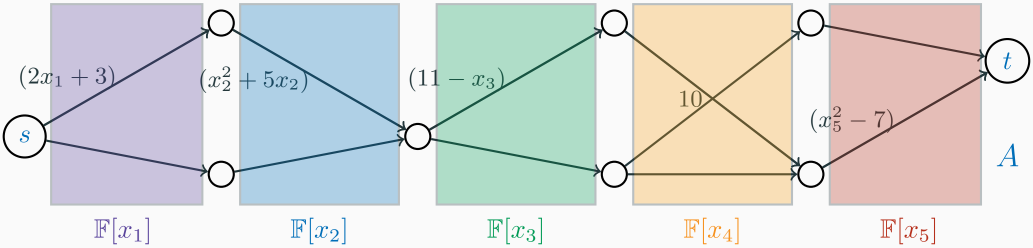
- On each s to t path, every variable is read-once, oblivious of the others.

Read-once Oblivious ABPs



ROABP (n-variate, degree-d, width-w)

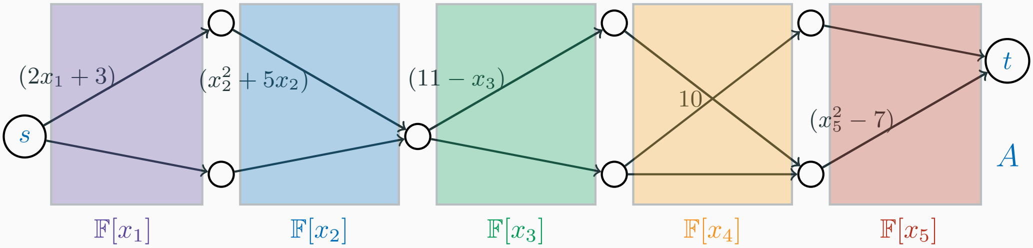
- On each s to t path, every variable is read-once, oblivious of the others.
- The i^{th} layer of edges only has degree-d univariates in x_i as labels.



ROABP (n-variate, degree-d, width-w)

- On each s to t path, every variable is read-once, oblivious of the others.
- The i^{th} layer of edges only has degree-d univariates in x_i as labels.
- Width of the ROABP: Maximum number of vertices in any layer.

Read-once Oblivious ABPs



ROABP (n-variate, degree-d, width-w, order σ)

- On each s to t path, every variable is read-once, oblivious of the others.
- The i^{th} layer of edges only has degree-d univariates in $x_{\sigma(i)}$ as labels.
- Width of the ROABP: Maximum number of vertices in any layer.
- Order of the ROABP: permutation $\sigma \in s_n$ in which the variables are read.

Example. Width depends on Order

$F(\overline{x}, \overline{y})$ has a width 2 ROABP in the order $(x_1, y_1, x_2, y_2, \ldots, x_n, y_n)$,
but requires width 2^n in the order $(x_1, x_2, \ldots, x_n, y_1, y_2, \ldots, y_n)$.

$$F(\overline{x}, \overline{y}) := (x_1 + y_1)(x_2 + y_2) \cdots (x_n + y_n)$$

State of the art for ROABPs

- **Lower Bounds.** Easy, due to an explicit characterization (Nisan 1991) to compute the optimal width in each layer exactly.

State of the art for ROABPs

- **Lower Bounds.** Easy, due to an explicit characterization (Nisan 1991) to compute the optimal width in each layer exactly.
- **Identity Testing.**
  - Algorithm can access circuit ("Whitebox"): Deterministic time $\text{poly}(n, d, w)$.
  - Algorithm can only query ("Blackbox"): Deterministic time $(ndw)^{O(\log n)}$.

State of the art for ROABPs

- **Lower Bounds.** Easy, due to an explicit characterization (Nisan 1991) to compute the optimal width in each layer exactly.
- **Identity Testing.**
  - Algorithm can access circuit ("Whitebox"): Deterministic time $\mathrm{poly}(n, d, w)$.
  - Algorithm can only query ("Blackbox"): Deterministic time $(ndw)^{O(\log n)}$.
- **Reconstruction.** Randomized polytime, and deterministic time $(ndw)^{O(\log n)}$, when the algorithm is given the order.

State of the art for ROABPs

- **Lower Bounds.** Easy, due to an explicit characterization (Nisan 1991) to compute the optimal width in each layer exactly.
- **Identity Testing.**
  - Algorithm can access circuit ("Whitebox"): Deterministic time $\mathrm{poly}(n, d, w)$.
  - Algorithm can only query ("Blackbox"): Deterministic time $(ndw)^{O(\log n)}$.
- **Reconstruction.** Randomized polytime, and deterministic time $(ndw)^{O(\log n)}$, when the algorithm is given the order.

Q. What is the complexity of finding the order?

The Question

Order Finding Problem

Given parameters $n, d, w \in \mathbb{N}$ and a polynomial $f(\overline{x})$, find some order σ in which f has an ROABP of width at most w.

The Question

Order Finding Problem

Given parameters $n, d, w \in \mathbb{N}$ and a polynomial $f(\overline{x})$, find some order σ in which f has an ROABP of width at most w.

Note. Using reconstruction, we can check if the given w is correct. So we can assume WLOG that f has an ROABP of width w in some order, hence the following is a simpler problem.

The Question

Order Finding Problem

Given parameters $n, d, w \in \mathbb{N}$ and a polynomial $f(\overline{x})$, find some order σ in which f has an ROABP of width at most w.

Note. Using reconstruction, we can check if the given w is correct. So we can assume WLOG that f has an ROABP of width w in some order, hence the following is a simpler problem.

Order Finding Problem (Decision)

Given an n-variate, degree-d polynomial $f(\overline{x})$, and a parameter $w \in \mathbb{N}$, determine if f has an ROABP of width at most w in some order σ.

Our Results

NP hardness (Algebraic Circuit Minimization)

Order finding problem is NP-hard, even when f is given as an algebraic circuit.

Our Results

NP hardness (Algebraic Circuit Minimization)

Order finding problem is NP-hard, even when f is given as an algebraic circuit.

NP hardness for constant degree (Algebraic MCSP)

For any constant $\Delta \geq 6$, order finding for n-variate, degree-Δ polynomials is NP-hard, even when f is given in the dense representation (algebraic analogue of a truth table).

Our Results

NP hardness (Algebraic Circuit Minimization)

Order finding problem is NP-hard, even when f is given as an algebraic circuit.

NP hardness for constant degree (Algebraic MCSP)

For any constant $\Delta \geq 6$, order finding for n-variate, degree-Δ polynomials is NP-hard, even when f is given in the dense representation (algebraic analogue of a truth table).

Average-case algorithm

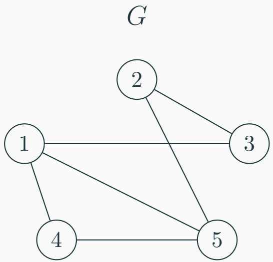
Randomized order-finding algorithm that runs in polytime for a random/generic ROABP.

Proof Ideas

NP-hardness

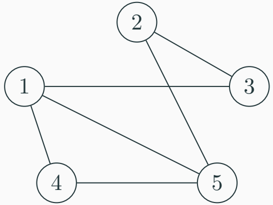
Goal. Show that finding an optimal order is hard.

Goal. Show that finding an optimal order is hard.



Goal. Show that finding an optimal order is hard.

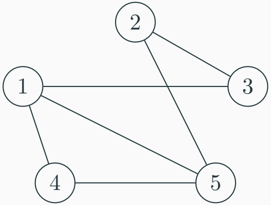
G



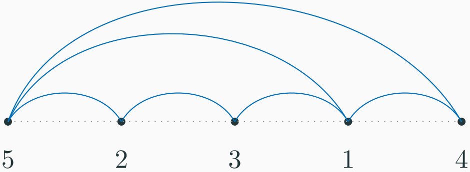
Arrangement of G with $(5, 2, 3, 1, 4)$

Goal. Show that finding an optimal order is hard.

G

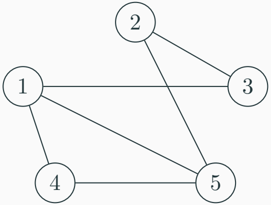


Arrangement of G with $(5, 2, 3, 1, 4)$

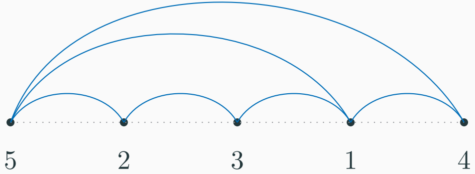


Goal. Show that finding an optimal order is hard.

G



Arrangement of G with $(5, 2, 3, 1, 4)$

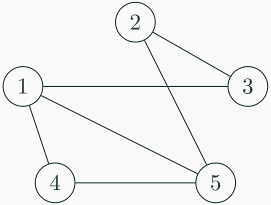


$$\text{CutWidth}(G) \;:=\; \min_{\sigma \in S_n} \text{CutWidth}_\sigma(G) \;:=\; \min_{\sigma \in S_n} \left(\max_{i \in [n]} \#\text{edges}(\sigma[1:i], \sigma[i+1:n]) \right)$$

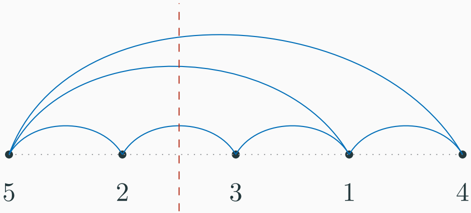
Linear Arrangement of Graphs

Goal. Show that finding an optimal order is hard.

G



CutWidth of G for $(5, 2, 3, 1, 4)$ is 3



$$\text{CutWidth}(G) \;:=\; \min_{\sigma \in S_n} \text{CutWidth}_\sigma(G) \;:=\; \min_{\sigma \in S_n} \left(\max_{i \in [n]} \#\text{edges}(\sigma[1:i], \sigma[i+1:n]) \right)$$

Reduction from `CutWidth`

$$\mathsf{CutWidth}(G) \;\; := \;\; \min_{\sigma \in S_n} \left(\max_{i \in [n]} \;\; \#\mathsf{edges}(\sigma[1:i], \sigma[i+1:n]) \right)$$

Reduction from CutWidth

$$\text{CutWidth}(G) := \min_{\sigma \in S_n} \left(\max_{i \in [n]} \#\text{edges}(\sigma[1:i], \sigma[i+1:n]) \right)$$

$$\text{ROABPwidth}(f) := \min_{\sigma \in S_n} \left(\max_{i \in [n]} \# \{\text{vertices in layer } i\} \right)$$

Reduction from CutWidth

$$\text{CutWidth}(G) \;:=\; \min_{\sigma \in S_n} \left(\max_{i \in [n]} \#\text{edges}(\sigma[1:i], \sigma[i+1:n]) \right)$$

$$\text{ROABPwidth}(f) \;:=\; \min_{\sigma \in S_n} \left(\max_{i \in [n]} \# \{\text{vertices in layer } i\} \right)$$

Theorem (implied by [Nisan 1991])

For any $f(\overline{x})$, the optimal ROABP for f in the order σ has exactly $w_i = \text{rk}\left(M_f^{(\sigma,i)}\right)$ vertices in layer i,

Reduction from CutWidth

$$\text{CutWidth}(G) \;:=\; \min_{\sigma \in S_n} \left(\max_{i \in [n]} \; \#\text{edges}(\sigma[1:i], \sigma[i+1:n]) \right)$$

$$\text{ROABPwidth}(f) \;:=\; \min_{\sigma \in S_n} \quad \left(\max_{i \in [n]} \; \# \left\{ \text{vertices in layer } i \right\} \right)$$

Theorem (implied by [Nisan 1991])

For any $f(\overline{x})$, the optimal ROABP for f in the order σ has exactly $w_i = \mathrm{rk}\left(M_f^{(\sigma,i)}\right)$ vertices in layer i, where $M_f^{(\sigma,i)}$ is as follows, for $\overline{x}_L = \left\{x_{\sigma(1)}, \ldots, x_{\sigma(i)}\right\}, \overline{x}_R = \left\{x_{\sigma(i+1)}, \ldots, x_{\sigma(n)}\right\}$.

$$\forall m \in \mathrm{mons}(\overline{x}_L), m' \in \mathrm{mons}(\overline{x}_R), \qquad M_f^{(\sigma,i)}[m, m'] = \mathrm{coeff}_f(m \cdot m')$$

Reduction from `CutWidth`

Lemma (Bhargava-Dutta-Ghosh-T. 2024)

Given any graph $G = (V, E)$, there is a polynomial $f_G(x_1, \ldots, x_n)$ such that:

- $n = |V|$,
- $\forall \sigma \in S_n, i \in [n], \quad \mathrm{rk}\left(M_{f_G}^{(\sigma,i)}\right) = \#\text{edges}(\sigma[1:i], \sigma[i+1:n]) + 2$,

Lemma (Bhargava-Dutta-Ghosh-T. 2024)

Given any graph $G = (V, E)$, there is a polynomial $f_G(x_1, \ldots, x_n)$ such that:

- $n = |V|$,
- $\forall \sigma \in S_n, i \in [n], \quad \mathrm{rk}\left(M_{f_G}^{(\sigma,i)}\right) = \#\text{edges}(\sigma[1:i], \sigma[i+1:n]) + 2$,
- $\deg(f_G) = 2 \cdot \text{degree}(G)$ and $\mathrm{ideg}(f_G) = \text{degree}(G) + 1$,
- f_G has $|E| + |V| + 1$ monomials.

Reduction from CutWidth

Lemma (Bhargava-Dutta-Ghosh-T. 2024)

Given any graph $G = (V, E)$, there is a polynomial $f_G(x_1, \ldots, x_n)$ such that:

- $n = |V|$,
- $\forall \sigma \in S_n, i \in [n], \quad \mathrm{rk}\left(M_{f_G}^{(\sigma,i)}\right) = \#\text{edges}(\sigma[1:i], \sigma[i+1:n]) + 2$,
- $\deg(f_G) = 2 \cdot \text{degree}(G)$ and $\mathrm{ideg}(f_G) = \text{degree}(G) + 1$,
- f_G has $|E| + |V| + 1$ monomials.

Fact (Monien-Sudborough 1988)

CutWidth is NP-complete, even for planar graphs of degree 3.

Theorem (Algebraic MCSP)

For any constant $\Delta \geq 6$, order finding for n-variate, degree-Δ polynomials is NP-hard, even when f is given in the dense representation (algebraic analogue of a truth table).

Proof. Truth table has length $\binom{n+\Delta}{\Delta} = \mathrm{poly}(n)$ for constant Δ.

Theorem (Algebraic MCSP)

For any constant $\Delta \geq 6$, order finding for n-variate, degree-Δ polynomials is NP-hard, even when f is given in the dense representation (algebraic analogue of a truth table).

Proof. Truth table has length $\binom{n+\Delta}{\Delta} = \mathrm{poly}(n)$ for constant Δ.

Theorem (Algebraic Circuit Minimization)

Order finding problem is NP-hard, even when f is given as an algebraic circuit.

Proof. $\mathrm{CircuitSize}(f_G) = O(n^3)$. (Truth table length $\sim 2^n$ for degree $\Omega(n)$.)

Proof Ideas

E-time worst-case algorithm

How to beat $n!$?

How to beat $n!$?

Theorem (Nisan's characterization)

ROABPwidth$_\sigma(f(x_1,\ldots,x_n)) \le w$, **iff** $\mathrm{rk}\left(M_f^{(\sigma,i)}\right) \le w$ for all $1 < i < n$.

E.g. For $n = 5$, $\sigma = (5,2,3,1,4)$, $\mathrm{rk}(M_f^{\{5\}})$, $\mathrm{rk}(M_f^{\{2,5\}})$, $\mathrm{rk}(M_f^{\{2,3,5\}})$ and $\mathrm{rk}(M_f^{\{1,2,3,5\}})$ are all at most w.

M_f^S has mons in x_S and $x_{\overline{S}}$ as rows and columns.

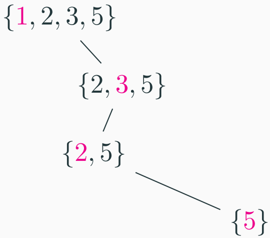
How to beat $n!$?

Theorem **(Nisan's characterization)**

$\text{ROABPwidth}_\sigma(f(x_1, \ldots, x_n)) \leq w$, **iff** $\text{rk}\left(M_f^{(\sigma,i)}\right) \leq w$ for all $1 < i < n$.

E.g. For $n = 5$, $\sigma = (5, 2, 3, 1, 4)$, $\text{rk}(M_f^{\{5\}})$, $\text{rk}(M_f^{\{2,5\}})$, $\text{rk}(M_f^{\{2,3,5\}})$ and $\text{rk}(M_f^{\{1,2,3,5\}})$ are all at most w.

M_f^S has mons in x_S and $x_{\overline{S}}$ as rows and columns.



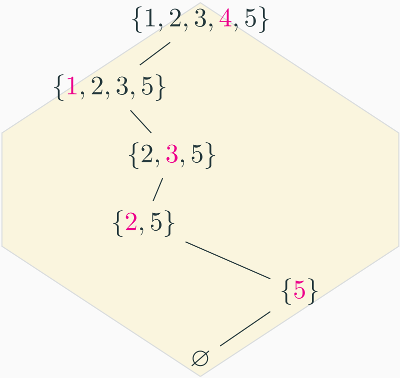
How to beat $n!$?

Theorem (Nisan's characterization)

$\text{ROABPwidth}_\sigma(f(x_1, \ldots, x_n)) \le w$, **iff** $\text{rk}\left(M_f^{(\sigma,i)}\right) \le w$ for all $1 < i < n$.

E.g. For $n = 5$, $\sigma = (5, 2, 3, 1, 4)$, $\text{rk}(M_f^{\{5\}})$, $\text{rk}(M_f^{\{2,5\}})$, $\text{rk}(M_f^{\{2,3,5\}})$ and $\text{rk}(M_f^{\{1,2,3,5\}})$ are all at most w.

M_f^S has mons in x_S and $x_{\overline{S}}$ as rows and columns.



How to beat $n!$?

Theorem **(Nisan's characterization)**

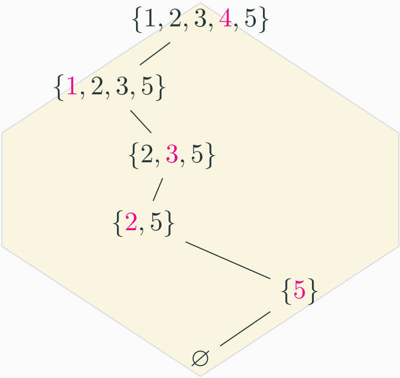
$\mathrm{ROABPwidth}_\sigma(f(x_1, \ldots, x_n)) \leq w$, **iff** $\mathrm{rk}\left(M_f^{(\sigma,i)}\right) \leq w$ for all $1 < i < n$.

E.g. For $n = 5$, $\sigma = (5, 2, 3, 1, 4)$, $\mathrm{rk}(M_f^{\{5\}})$, $\mathrm{rk}(M_f^{\{2,5\}})$, $\mathrm{rk}(M_f^{\{2,3,5\}})$ and $\mathrm{rk}(M_f^{\{1,2,3,5\}})$ are all at most w.

M_f^S has mons in x_S and $x_{\overline{S}}$ as rows and columns.

Observation. $\mathrm{ROABPwidth}_\sigma(f(x_1, \ldots, x_n)) \leq w$ **iff** 'σ traces an $\varnothing$ to $[n]$ path in the graph $H_w(f)$'.

> $H_w(f)$: induced subgraph of hypercube, where $S \in H_w(f)$ if and only if $\mathrm{rk}(M_f^S) \leq w$.



E-time algorithm: Search on the Hypercube

Observation

ROABPwidth$_\sigma(f(x_1, \ldots, x_n)) \leq w$ **iff** σ traces an $\varnothing$ to $[n]$ path in the graph $H_w(f)$.

E-time algorithm: Search on the Hypercube

Observation

ROABPwidth$_\sigma(f(x_1, \ldots, x_n)) \leq w$ **iff** σ traces an $\varnothing$ to $[n]$ path in the graph $H_w(f)$.

Fact

For any $f(x_1, \ldots, x_n)$ of deg d, and $S \subseteq [n]$, checking if $\mathrm{rk}(M_f^S) \leq w$ reduces to PIT.

E-time algorithm: Search on the Hypercube

Observation

ROABPwidth$_\sigma(f(x_1,\ldots,x_n)) \leq w$ **iff** σ traces an $\varnothing$ to $[n]$ path in the graph $H_w(f)$.

Fact

For any $f(x_1,\ldots,x_n)$ of deg d, and $S \subseteq [n]$, checking if $\mathrm{rk}(M_f^S) \leq w$ reduces to PIT.

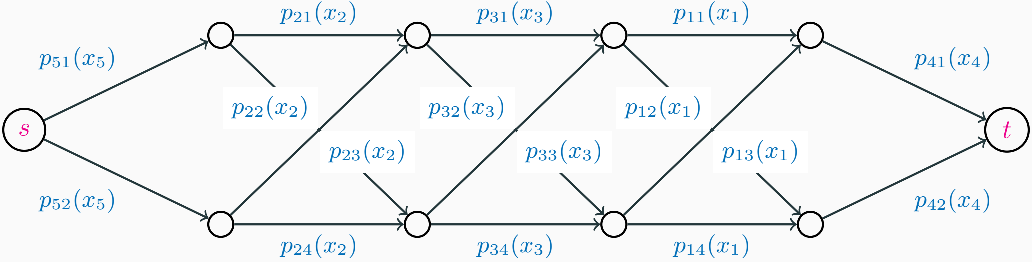
Algorithm. `FindOrder(f,w)`

1. `PopulateGraph(f,w)`: Find $H_w(f)$ using a DFS starting at $\varnothing$ (and above fact).
2. Output any σ that traces an $\varnothing$ to $[n]$ path in $H_w(f)$.

Proof Ideas

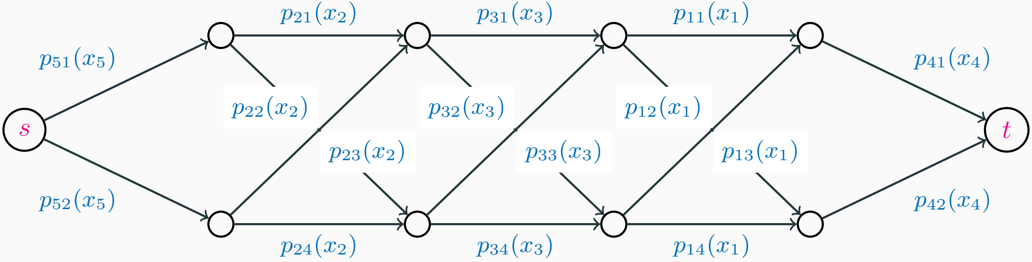
Algorithm for the generic case

Random/generic ROABPs



Generic ROABP for $n = 5$, $w = 2$ and $\sigma = (5, 2, 3, 1, 4)$: random coeffs for p_{ij}s.

Random/generic ROABPs



Generic ROABP for $n = 5$, $w = 2$ and $\sigma = (5, 2, 3, 1, 4)$: random coeffs for p_{ij}s.

Definition **($(n, d, w, \sigma, \mathcal{D})$-Generic ROABP)**

ROABP in order σ with all coefficients of edge labels ($\sim ndw^2$) iid according to $\mathcal{D}$.

Bad inputs for `PopulateGraph`

Algorithm. `FindOrder(f,w)`

1. `PopulateGraph(f,w)`: Find $H_w(f)$ using a DFS starting at $\varnothing$ (and above fact).
2. Output any σ that traces an $\varnothing$ to $[n]$ path in the populated graph.

Bad inputs for `PopulateGraph`

Algorithm. `FindOrder(f,w)`

1. `PopulateGraph(f,w)`: Find $H_w(f)$ using a DFS starting at $\varnothing$ (and above fact).
2. Output any σ that traces an $\varnothing$ to $[n]$ path in the populated graph.

Key idea. Bad inputs are special (i.e. not generic).

Bad inputs for `PopulateGraph`

Algorithm. `FindOrder(f,w)`

1. `PopulateGraph(f,w)`: Find $H_w(f)$ using a DFS starting at $\varnothing$ (and above fact).
2. Output any σ that traces an $\varnothing$ to $[n]$ path in the populated graph.

Key idea. Bad inputs are special (i.e. not generic).

- Bad input f: $H_w(f)$ has many vertices, but very few $\varnothing$ to $[n]$ paths.
  DFS has to backtrack from several blocked paths.

Bad inputs for `PopulateGraph`

Algorithm. `FindOrder(f,w)`

1. `PopulateGraph(f,w)`: Find $H_w(f)$ using a DFS starting at $\varnothing$ (and above fact).
2. Output any σ that traces an $\varnothing$ to $[n]$ path in the populated graph.

Key idea. Bad inputs are special (i.e. not generic).

- Bad input f: $H_w(f)$ has many vertices, but very few $\varnothing$ to $[n]$ paths.
  DFS has to backtrack from several blocked paths.
- We show: for generic $f \in \text{ROABP}(n,d,w,\sigma)$, $H_w(f)$ only has the obvious vertices.
  $\mathrm{rk}(M_f^S) \leq w$ when S is a prefix of σ, or $|S|$ is too small (M_f^S is skewed).

Bad inputs for `PopulateGraph`

Algorithm. `FindOrder(f,w)`

1. `PopulateGraph(f,w)`: Find $H_w(f)$ using a DFS starting at $\varnothing$ (and above fact).
2. Output any σ that traces an $\varnothing$ to $[n]$ path in the populated graph.

Key idea. Bad inputs are special (i.e. not generic).

- Bad input f: $H_w(f)$ has many vertices, but very few $\varnothing$ to $[n]$ paths.
  DFS has to backtrack from several blocked paths.
- We show: for generic $f \in \text{ROABP}(n, d, w, \sigma)$, $H_w(f)$ only has the obvious vertices.
  $\text{rk}(M_f^S) \leq w$ when S is a prefix of σ, or $|S|$ is too small (M_f^S is skewed).
- For "inconsistent" S, fs with $\text{rk}(M_f^S) \leq w$ form a strict subvariety of $\text{ROABP}(n, d, w, \sigma)$.
  [SZ lemma]: $H_w(f)$ has $n^{O(\log_d(w))}$ vertices w.h.p., for any large-enough domain.

Theorem (Average-case algorithm)

Over all sets D of size 2^{10n},and for any n, d, w, σ,
`PopulateGraph` runs in randomized time $n^{O(\log_d(w))} \cdot \mathrm{poly}(d, w)$ on a random/generic input from ROABP(n, d, w, σ) w.h.p., where the coeffs are drawn from D.

- Polynomial time when $w = d^{O(1)}$.
- Quasi-polynomial time even when $d = O(1)$.

Average Case Results

Theorem (Average-case algorithm)

Over all sets D of size 2^{10n},and for any n, d, w, σ,
`PopulateGraph` runs in randomized time $n^{O(\log_d(w))} \cdot \mathrm{poly}(d, w)$ on a random/generic input from $\mathrm{ROABP}(n, d, w, \sigma)$ w.h.p., where the coeffs are drawn from D.

- Polynomial time when $w = d^{O(1)}$.
- Quasi-polynomial time even when $d = O(1)$.

Remark. We need $|D| \sim 2^n$ due to a union bound over all inconsistent S.

Summary

Our Work

- ROABPs can be proper-learnt efficiently when order is known.

Our Work

- ROABPs can be proper-learnt efficiently when order is known.
- Order-finding problem is NP-hard even in simple, "white-box" settings.
  (Only other algebraic MCSP results are for tensor and Waring ranks [Håstad'90].)

Our Work

- ROABPs can be proper-learnt efficiently when order is known.
- Order-finding problem is NP-hard even in simple, "white-box" settings.
  (Only other algebraic MCSP results are for tensor and Waring ranks [Håstad'90].)
- Order-finding can be solved in average case in (quasi-)polynomial time.

Our Work

- ROABPs can be proper-learnt efficiently when order is known.
- Order-finding problem is NP-hard even in simple, “white-box” settings.
  (Only other algebraic MCSP results are for tensor and Waring ranks [Håstad’90].)
- Order-finding can be solved in average case in (quasi-)polynomial time.
- **Approximation algorithms.**
  - ROABPwidth is hard to approximate up to any constant factor under SSE conjecture.
  - Unconditionally, any constant approximation for ROABPwidth leads to a PTAS.

Open Questions

- **Average-case algorithm.**
  - Polynomial time for constant individual degree?

    Will require a different approach.
  - Better dependence on domain-size.

    Different argument that bypasses the union bound.

Open Questions

- **Average-case algorithm.**
  - Polynomial time for constant individual degree?

    Will require a different approach.
  - Better dependence on domain-size.

    Different argument that bypasses the union bound.

- **Hardness of approximation.**
  - Is `CutWidth` hard to approximate up to a constant factor (without SSE)?
  - Is ROABPwidth hard to approximate (for some other reason)?

Thank you!

ABP-figure credits: Prerona Chatterjee