Demystifying the border of depth-3 algebraic circuits

Accepted to the 62^{nd} IEEE Symposium on Foundations of Computer Science (FOCS 2021).

Join work with Prateek Dwivedi (IIT Kanpur) & Nitin Saxena (IIT Kanpur).

Pranjal Dutta
Google Ph.D. Fellow, CMI & Visiting Research Scholar, IIT Kanpur

11^{th} November, 2021

Oxford-Warwick Complexity Seminar

Table of Contents

Algebraic Complexity Theory

Basic goal

- ❑ P **versus** NP**.** Proving $\mathsf{P} \neq \mathsf{NP}$ is one of the most fundamental open problems at the intersection of theoretical computer science and mathematics.

Basic goal

- P **versus** NP. Proving $\mathsf{P} \neq \mathsf{NP}$ is one of the most fundamental open problems at the intersection of theoretical computer science and mathematics.
- Very few techniques are known that could potentially break the 1994 Razborov-Rudich *'natural proofs barrier'*.

Basic goal

- P **versus** NP. Proving $\mathsf{P} \neq \mathsf{NP}$ is one of the most fundamental open problems at the intersection of theoretical computer science and mathematics.
- Very few techniques are known that could potentially break the 1994 Razborov-Rudich *'natural proofs barrier'*.
- In 2001 Mulmuley and Sohoni published **Geometric Complexity Theory 1** (GCT1) in which they describe a method that could potentially break the barrier.

Basic goal

- P **versus** NP**.** Proving $\mathsf{P} \neq \mathsf{NP}$ is one of the most fundamental open problems at the intersection of theoretical computer science and mathematics.
- Very few techniques are known that could potentially break the 1994 Razborov-Rudich *'natural proofs barrier'*.
- In 2001 Mulmuley and Sohoni published **Geometric Complexity Theory 1** (GCT1) in which they describe a method that could potentially break the barrier.
  - It is built on Valiant's algebraic complexity theory framework (1979) to prove the algebraic $\mathsf{P} \neq \mathsf{NP}$, namely $\mathsf{VP} \neq \mathsf{VNP}$.

Basic goal

- P **versus** NP. Proving P $\neq$ NP is one of the most fundamental open problems at the intersection of theoretical computer science and mathematics.
- Very few techniques are known that could potentially break the 1994 Razborov-Rudich *'natural proofs barrier'*.
- In 2001 Mulmuley and Sohoni published **Geometric Complexity Theory 1** (GCT1) in which they describe a method that could potentially break the barrier.
  - It is built on Valiant's algebraic complexity theory framework (1979) to prove the algebraic P $\neq$ NP, namely VP $\neq$ VNP.
  - It defines **Border Complexity**, which was independently defined by Bürgisser (2001).

Basic goal

- P **versus** NP**.** Proving $\mathsf{P} \neq \mathsf{NP}$ is one of the most fundamental open problems at the intersection of theoretical computer science and mathematics.
- Very few techniques are known that could potentially break the 1994 Razborov-Rudich *'natural proofs barrier'*.
- In 2001 Mulmuley and Sohoni published **Geometric Complexity Theory 1** (GCT1) in which they describe a method that could potentially break the barrier.
  - It is built on Valiant's algebraic complexity theory framework (1979) to prove the algebraic $\mathsf{P} \neq \mathsf{NP}$, namely $\mathsf{VP} \neq \mathsf{VNP}$.
  - It defines **Border Complexity**, which was independently defined by Bürgisser (2001). We will consider 'algebraic' notion of border complexity.

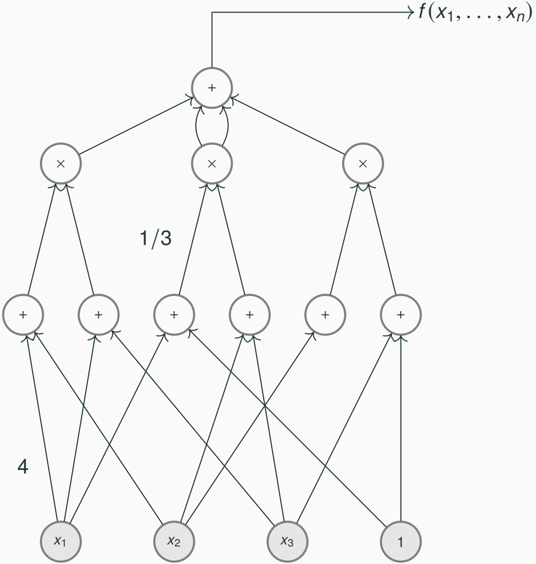
Basic goal

- P **versus** NP. Proving $\mathsf{P} \neq \mathsf{NP}$ is one of the most fundamental open problems at the intersection of theoretical computer science and mathematics.
- Very few techniques are known that could potentially break the 1994 Razborov-Rudich *'natural proofs barrier'*.
- In 2001 Mulmuley and Sohoni published **Geometric Complexity Theory 1** (GCT1) in which they describe a method that could potentially break the barrier.
  - It is built on Valiant's algebraic complexity theory framework (1979) to prove the algebraic $\mathsf{P} \neq \mathsf{NP}$, namely $\mathsf{VP} \neq \mathsf{VNP}$.
  - It defines **Border Complexity**, which was independently defined by Bürgisser (2001). We will consider 'algebraic' notion of border complexity.
  - It proposes to prove border complexity lower bounds using representation theory, which is developed further in [GCT2, Mulmuley-Sohoni'08].

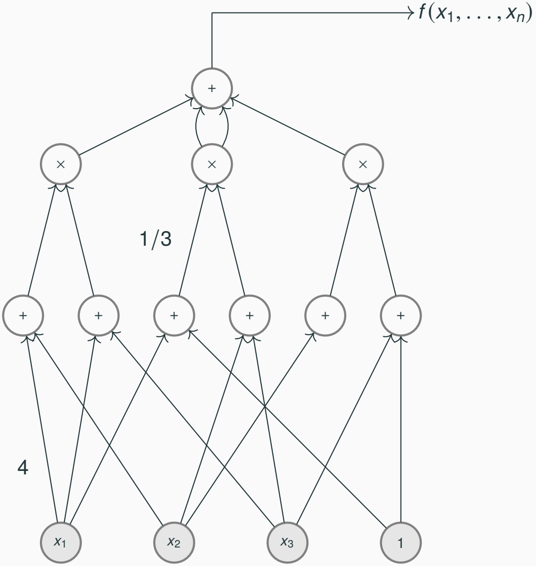
Basic goal

- P **versus** NP. Proving $\mathsf{P} \neq \mathsf{NP}$ is one of the most fundamental open problems at the intersection of theoretical computer science and mathematics.
- Very few techniques are known that could potentially break the 1994 Razborov-Rudich *'natural proofs barrier'*.
- In 2001 Mulmuley and Sohoni published **Geometric Complexity Theory 1** (GCT1) in which they describe a method that could potentially break the barrier.
  - It is built on Valiant's algebraic complexity theory framework (1979) to prove the algebraic $\mathsf{P} \neq \mathsf{NP}$, namely $\mathsf{VP} \neq \mathsf{VNP}$.
  - It defines **Border Complexity**, which was independently defined by Bürgisser (2001). We will consider 'algebraic' notion of border complexity.
  - It proposes to prove border complexity lower bounds using representation theory, which is developed further in [GCT2, Mulmuley-Sohoni'08].
- [$\mathsf{P} \stackrel{?}{=} \mathsf{NP}$, Aronson 2011] calls GCT *"The String Theory of Computer Science"*.

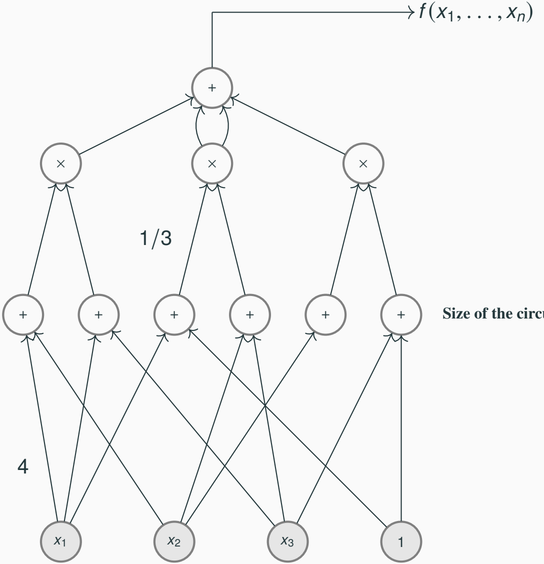
Algebraic circuits



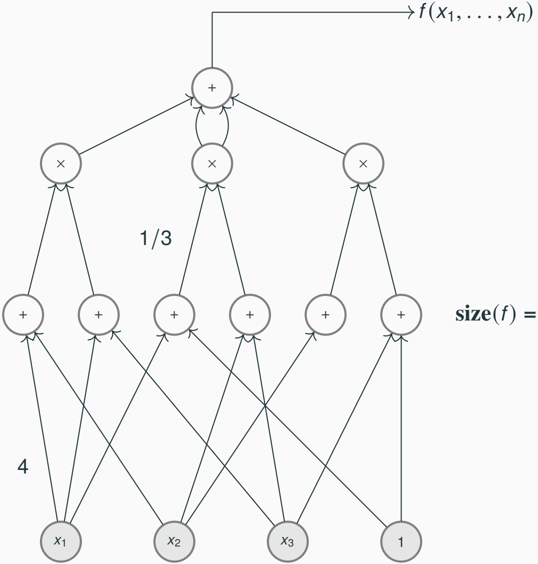
Algebraic circuits



Algebraic circuits

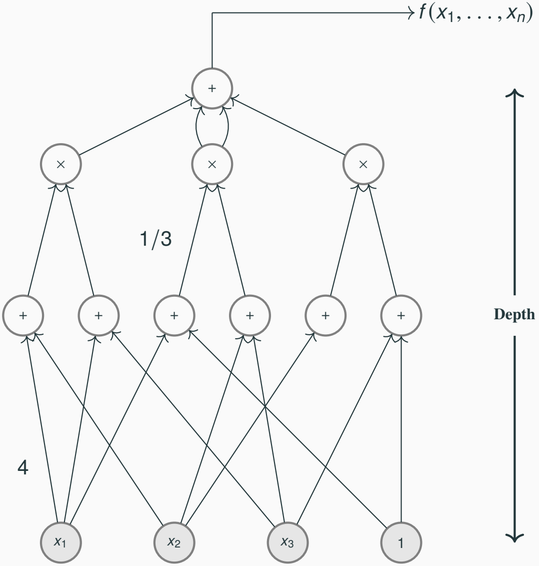


Size of the circuit = number of nodes + edges



size(f) = min size of the circuit computing f

Algebraic circuits



Computationally ‘easy’ polynomials

Computationally ‘easy’ polynomials

VP = “easy to compute” [Valiant’79]

The class VP is defined as the set of all sequences of polynomials $(f_n(x_1, \ldots, x_n))_{n \geq 1}$ such that $\mathrm{size}(f_n), \mathsf{deg}(f_n)$ are both bounded by n^c for some constant c.

Computationally 'easy' polynomials

VP = "easy to compute" [Valiant'79]

The class VP is defined as the set of all sequences of polynomials $(f_n(x_1, \ldots, x_n))_{n \geq 1}$ such that $\mathrm{size}(f_n), \mathsf{deg}(f_n)$ are both bounded by n^c for some constant c.

Examples:

- $f_n := x_1 \cdots x_n$.

Computationally 'easy' polynomials

VP = "easy to compute" [Valiant'79]

The class VP is defined as the set of all sequences of polynomials $(f_n(x_1, \ldots, x_n))_{n \geq 1}$ such that $\mathrm{size}(f_n), \mathsf{deg}(f_n)$ are both bounded by n^c for some constant c.

Examples:

- $f_n \; := \; x_1 \cdots x_n.$
- $f_n \; := \; x_1^n + \ldots + x_n^n.$

Computationally 'easy' polynomials

VP = "easy to compute" [Valiant'79]

The class VP is defined as the set of all sequences of polynomials $(f_n(x_1, \ldots, x_n))_{n \geq 1}$ such that $\mathrm{size}(f_n), \deg(f_n)$ are both bounded by n^c for some constant c.

Examples:

- $f_n := x_1 \cdots x_n$.
- $f_n := x_1^n + \ldots + x_n^n$.
- $f_n := \sum_{S \subseteq [n]} \prod_{j \in S} x_j$

Computationally 'easy' polynomials

VP = "easy to compute" [Valiant'79]

The class VP is defined as the set of all sequences of polynomials $(f_n(x_1, \ldots, x_n))_{n \geq 1}$ such that $\mathrm{size}(f_n), \mathsf{deg}(f_n)$ are both bounded by n^c for some constant c.

Examples:

- $f_n := x_1 \cdots x_n$.
- $f_n := x_1^n + \ldots + x_n^n$.
- $f_n := \sum_{S \subseteq [n]} \prod_{j \in S} x_j = \prod_{i=1}^{n} (1 + x_i)$.

The determinant polynomial

The determinant polynomial

- Let $X_n = [x_{i,j}]_{1 \le i,j \le n}$ be a $n \times n$ matrix of distinct variables $x_{i,j}$. Let $S_n := \{\pi \mid \pi : \{1, \ldots, n\} \longrightarrow \{1, \ldots, n\}$ such that π is bijective $\}$. Define

$$f_n \;:=\; \det(X_n) \;=\; \sum_{\pi \in S_n} \mathrm{sgn}(\pi) \prod_{i=1}^{n} x_{i,\pi(i)} \;.$$

The determinant polynomial

- Let $X_n = [x_{i,j}]_{1 \leq i,j \leq n}$ be a $n \times n$ matrix of distinct variables $x_{i,j}$. Let $S_n := \{\pi \mid \pi : \{1, \ldots, n\} \longrightarrow \{1, \ldots, n\} \text{ such that } \pi \text{ is bijective } \}$. Define

$$f_n := \det(X_n) = \sum_{\pi \in S_n} \operatorname{sgn}(\pi) \prod_{i=1}^{n} x_{i,\pi(i)} .$$

- $\det$ is *universal*, i.e. any polynomial $f(\boldsymbol{x})$ can be computed as a determinant of a square matrix whose entries are polynomials of degree ≤ 1.

The determinant polynomial

- Let $X_n = [x_{i,j}]_{1 \leq i,j \leq n}$ be a $n \times n$ matrix of distinct variables $x_{i,j}$. Let $S_n := \{\pi \mid \pi : \{1, \dots, n\} \longrightarrow \{1, \dots, n\}$ such that π is bijective $\}$. Define

$$f_n := \det(X_n) = \sum_{\pi \in S_n} \operatorname{sgn}(\pi) \prod_{i=1}^{n} x_{i,\pi(i)} .$$

- $\det$ is *universal*, i.e. any polynomial $f(\boldsymbol{x})$ can be computed as a determinant of a square matrix whose entries are polynomials of degree ≤ 1.
- The minimum dimension of the matrix to compute f, is called the **determinantal complexity** $\mathrm{dc}(f)$.

The determinant polynomial

- Let $X_n = [x_{i,j}]_{1 \le i,j \le n}$ be a $n \times n$ matrix of distinct variables $x_{i,j}$. Let $S_n := \{\pi \mid \pi : \{1,\ldots,n\} \longrightarrow \{1,\ldots,n\}$ such that π is bijective $\}$. Define

$$f_n := \det(X_n) = \sum_{\pi \in S_n} \operatorname{sgn}(\pi) \prod_{i=1}^{n} x_{i,\pi(i)} .$$

- det is *universal*, i.e. any polynomial $f(\boldsymbol{x})$ can be computed as a determinant of a square matrix whose entries are polynomials of degree ≤ 1.

- The minimum dimension of the matrix to compute f, is called the **determinantal complexity** $\mathrm{dc}(f)$.

- E.g. $\mathrm{dc}(x_1 \cdots x_n) = n$, since

$$x_1 \cdots x_n = \det \begin{pmatrix} x_1 & 0 & \ldots & 0 \\ 0 & x_2 & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \ldots & x_n \end{pmatrix} .$$

Another ‘easy’ class VBP

- VBP: The class VBP is defined as the set of all sequences of polynomials $(f_n)_n$ with polynomially bounded $\mathsf{dc}(f_n)$.

Another 'easy' class VBP

- ❑ VBP: The class VBP is defined as the set of all sequences of polynomials $(f_n)_n$ with polynomially bounded $\mathsf{dc}(f_n)$.
- ❑ $\mathsf{VBP} \subseteq \mathsf{VP}$. It is open whether $\mathsf{VBP} \stackrel{?}{=} \mathsf{VP}$.

Another 'easy' class VBP

- VBP: The class VBP is defined as the set of all sequences of polynomials $(f_n)_n$ with polynomially bounded $\mathsf{dc}(f_n)$.
- $\mathsf{VBP} \subseteq \mathsf{VP}$. It is open whether $\mathsf{VBP} \stackrel{?}{=} \mathsf{VP}$.
- Often we will say f has a small ABP. This just means $\mathsf{dc}(f)$ is small.

Another 'easy' class VBP

- VBP: The class VBP is defined as the set of all sequences of polynomials $(f_n)_n$ with polynomially bounded $\mathsf{dc}(f_n)$.
- $\mathsf{VBP} \subseteq \mathsf{VP}$. It is open whether $\mathsf{VBP} \stackrel{?}{=} \mathsf{VP}$.
- Often we will say f has a small ABP. This just means $\mathsf{dc}(f)$ is small.
- Connections: Linear algebra, Volume, counting planar matchings.

‘Hard’ polynomials?

‘Hard’ polynomials?

- Are there hard polynomial families $(f_n)_n$ such that it cannot be computed by an n^c-size circuit, for *every* constant c? i.e. $\mathrm{size}(f_n) = n^{\omega(1)}$?

'Hard' polynomials?

- Are there hard polynomial families $(f_n)_n$ such that it cannot be computed by an n^c-size circuit, for *every* constant c? i.e. $\mathrm{size}(f_n) = n^{\omega(1)}$?
- A *random* polynomial with 0-1 coefficient is **hard** [Hrubeš-Yehudayoff ToC'11].

'Hard' polynomials?

- Are there hard polynomial families $(f_n)_n$ such that it cannot be computed by an n^c-size circuit, for *every* constant c? i.e. $\text{size}(f_n) = n^{\omega(1)}$?

- A *random* polynomial with 0-1 coefficient is **hard** [Hrubeš-Yehudayoff ToC'11].
  Challenge: Find an **explicit** one!

‘Hard’ polynomials?

- Are there hard polynomial families $(f_n)_n$ such that it cannot be computed by an n^c-size circuit, for *every* constant c? i.e. $\mathrm{size}(f_n) = n^{\omega(1)}$?
- A *random* polynomial with 0-1 coefficient is **hard** [Hrubeš-Yehudayoff ToC’11]. Challenge: Find an **explicit** one!
- Candidate hard polynomial:

$$\mathrm{perm}(X_n) = \sum_{\pi \in S_n} \prod_{i=1}^{n} x_{i,\pi(i)} .$$

‘Hard’ polynomials?

- Are there hard polynomial families $(f_n)_n$ such that it cannot be computed by an n^c-size circuit, for *every* constant c? i.e. $\text{size}(f_n) = n^{\omega(1)}$?
- A *random* polynomial with 0-1 coefficient is **hard** [Hrubeš-Yehudayoff ToC’11]. Challenge: Find an **explicit** one!
- Candidate hard polynomial:

$$\text{perm}(X_n) = \sum_{\pi \in S_n} \prod_{i=1}^{n} x_{i,\pi(i)} \ .$$

- perm is *universal*, i.e. any polynomial $f(\boldsymbol{x})$ can be computed as a permanent of a square matrix whose entries are polynomials of degree ≤ 1.

'Hard' polynomials?

- Are there hard polynomial families $(f_n)_n$ such that it cannot be computed by an n^c-size circuit, for *every* constant c? i.e. $\mathrm{size}(f_n) = n^{\omega(1)}$?

- A *random* polynomial with 0-1 coefficient is **hard** [Hrubeš-Yehudayoff ToC'11]. Challenge: Find an **explicit** one!

- Candidate hard polynomial:

$$\mathrm{perm}(X_n) \;=\; \sum_{\pi \in S_n} \prod_{i=1}^{n} x_{i,\pi(i)}\;.$$

- perm is *universal*, i.e. any polynomial $f(\boldsymbol{x})$ can be computed as a permanent of a square matrix whose entries are polynomials of degree ≤ 1.

- The minimum dimension of the matrix to compute f, is called the **permanental complexity** $\mathrm{pc}(f)$.

Valiant's Conjecture

Valiant's Conjecture

VNP = "hard to compute?" [Valiant 1979]

The class VNP is defined as the set of all sequences of polynomials $(f_n(x_1, \ldots, x_n))_{n \geq 1}$ such that $\mathsf{pc}(f_n)$is bounded by n^c for some constant c.

Valiant's Conjecture

VNP = "hard to compute?" [Valiant 1979]

The class VNP is defined as the set of all sequences of polynomials $(f_n(x_1, \dots, x_n))_{n\geq 1}$ such that $\mathsf{pc}(f_n)$is bounded by n^c for some constant c.

- Connections: Enumeration, counting matchings, Bosons etc.

Valiant's Conjecture

VNP = "hard to compute?" [Valiant 1979]

The class VNP is defined as the set of all sequences of polynomials $(f_n(x_1, \ldots, x_n))_{n \geq 1}$ such that $\mathsf{pc}(f_n)$is bounded by n^c for some constant c.

- Connections: Enumeration, counting matchings, Bosons etc.
- $\mathsf{VBP} \subseteq \mathsf{VP} \subseteq \mathsf{VNP}$.

Valiant's Conjecture

VNP = "hard to compute?" [Valiant 1979]

The class VNP is defined as the set of all sequences of polynomials $(f_n(x_1, \ldots, x_n))_{n \geq 1}$ such that $\mathsf{pc}(f_n)$is bounded by n^c for some constant c.

- Connections: Enumeration, counting matchings, Bosons etc.
- $\mathsf{VBP} \subseteq \mathsf{VP} \subseteq \mathsf{VNP}$.

Valiant's Conjecture [Valiant 1979]

$\mathsf{VBP} \neq \mathsf{VNP}$ & $\mathsf{VP} \neq \mathsf{VNP}$. Equivalently, $\mathsf{dc}(\mathsf{perm}_n)$ and $\mathsf{size}(\mathsf{perm}_n)$ are both $n^{\omega(1)}$.

Connections to Boolean circuit complexity

Connections to Boolean circuit complexity

- Separating algebraic classes are "easier" than separating classes in Boolean complexity [Bürgisser 1998]:

Connections to Boolean circuit complexity

- Separating algebraic classes are "easier" than separating classes in Boolean complexity [Bürgisser 1998]:
  - $\mathsf{P/poly} \neq \mathsf{NP/poly} \implies \mathsf{VBP} \neq \mathsf{VNP}$ and $\mathsf{VP} \neq \mathsf{VNP}$ (over finite fields).

Connections to Boolean circuit complexity

- Separating algebraic classes are "easier" than separating classes in Boolean complexity [Bürgisser 1998]:
  - $\mathsf{P/poly} \neq \mathsf{NP/poly} \implies \mathsf{VBP} \neq \mathsf{VNP}$ and $\mathsf{VP} \neq \mathsf{VNP}$ (over finite fields).
  - Assuming GRH (Generalized Riemann hypothesis), the results hold over $\mathbb{C}$ as well.

Summary

Summary

- A recent breakthrough. [Limaye-Srinivasan-Tavenas FOCS 2021] showed the *first super*polynomial lower bound for general **constant-depth** algebraic circuits!

Summary

- A recent breakthrough. [Limaye-Srinivasan-Tavenas FOCS 2021] showed the *first super*polynomial lower bound for general **constant-depth** algebraic circuits!

- Can there be 'algebraic natural proofs' to prove $\mathsf{VP} \neq \mathsf{VNP}$? Some answers: [Chatterjee-Kumar-Ramya-Saptharishi-Tengse 2020, Kumar-Ramya-Saptharishi-Tengse 2020].

Border Complexity and GCT

Waring rank

Waring rank

- Can ‘approximations’ also help in algebraic computational models?

Waring rank

- Can ‘approximations’ also help in algebraic computational models?
- An important measure is **Waring rank**, denoted $\mathsf{WR}(\cdot)$.

Waring rank

- ❑ Can 'approximations' also help in algebraic computational models?
- ❑ An important measure is **Waring rank**, denoted $\mathrm{WR}(\cdot)$.

Waring Rank

The smallest r such that a *homogeneous* degree d polynomial h can be written as a sum of d-th power of linear forms ℓ_i, i.e. $h = \sum_{i=1}^{r} \ell_i^d$.

Waring rank

- Can 'approximations' also help in algebraic computational models?
- An important measure is **Waring rank**, denoted $\mathsf{WR}(\cdot)$.

Waring Rank

The smallest r such that a *homogeneous* degree d polynomial h can be written as a sum of d-th power of linear forms ℓ_i, i.e. $h = \sum_{i=1}^{r} \ell_i^d$.

- Recall: $h = \sum_{e_1,..,e_n} a_{e_1,\ldots,e_n} x_1^{e_1} \cdots x_n^{e_n}$, is called **homogeneous** degree d polynomial if $\sum e_i = d$, for every tupple $(e_1, \ldots, e_n)$ such that $a_{e_1,\ldots,e_n} \neq 0$.

Waring rank

- Can 'approximations' also help in algebraic computational models?
- An important measure is **Waring rank**, denoted $\mathsf{WR}(\cdot)$.

Waring Rank

The smallest r such that a *homogeneous* degree d polynomial h can be written as a sum of d-th power of linear forms ℓ_i, i.e. $h = \sum_{i=1}^{r} \ell_i^d$.

- Recall: $h = \sum_{e_1,..,e_n} a_{e_1,...,e_n} x_1^{e_1} \cdots x_n^{e_n}$, is called **homogeneous** degree d polynomial if $\sum e_i = d$, for every tupple $(e_1, \ldots, e_n)$ such that $a_{e_1,...,e_n} \neq 0$.
- Recall: A linear form ℓ is of the form $a_1 x_1 + \ldots + a_n x_n$.

Waring rank

- Can 'approximations' also help in algebraic computational models?
- An important measure is **Waring rank**, denoted $\mathsf{WR}(\cdot)$.

Waring Rank

The smallest r such that a *homogeneous* degree d polynomial h can be written as a sum of d-th power of linear forms ℓ_i, i.e. $h = \sum_{i=1}^{r} \ell_i^d$.

- Recall: $h = \sum_{e_1,..,e_n} a_{e_1,\ldots,e_n} x_1^{e_1} \cdots x_n^{e_n}$, is called **homogeneous** degree d polynomial if $\sum e_i = d$, for every tupple $(e_1, \ldots, e_n)$ such that $a_{e_1,\ldots,e_n} \neq 0$.
- Recall: A linear form ℓ is of the form $a_1x_1 + \ldots + a_nx_n$.
- For any homogeneous polynomial h, $\mathsf{WR}(h)$ is *finite*.

Waring rank

- Can ‘approximations’ also help in algebraic computational models?
- An important measure is **Waring rank**, denoted $\mathsf{WR}(\cdot)$.

Waring Rank

The smallest r such that a *homogeneous* degree d polynomial h can be written as a sum of d-th power of linear forms ℓ_i, i.e. $h = \sum_{i=1}^{r} \ell_i^d$.

- Recall: $h = \sum_{e_1,..,e_n} a_{e_1,...,e_n} x_1^{e_1} \cdots x_n^{e_n}$, is called **homogeneous** degree d polynomial if $\sum e_i = d$, for every tupple $(e_1, \ldots, e_n)$ such that $a_{e_1,...,e_n} \neq 0$.
- Recall: A linear form ℓ is of the form $a_1x_1 + \ldots + a_nx_n$.
- For any homogeneous polynomial h, $\mathsf{WR}(h)$ is *finite*.
- $\mathsf{WR}(h) \leq r$ is denoted as $h \in \Sigma^{[r]} \wedge \Sigma$ (homogeneous *depth-3 diagonal* circuits).

Separating polynomial

Separating polynomial

- Let us try to *characterize* the bivariate degree 2 polynomials $h(x, y)$ such that $\mathsf{WR}(h) = 1$.

Separating polynomial

- Let us try to *characterize* the bivariate degree 2 polynomials $h(x, y)$ such that $\mathrm{WR}(h) = 1$.
- Let $h(x, y) = ax^2 + bxy + cy^2$.

Separating polynomial

- Let us try to *characterize* the bivariate degree 2 polynomials $h(x, y)$ such that $\mathrm{WR}(h) = 1$.
- Let $h(x, y) = ax^2 + bxy + cy^2$.
- $X_1 = \{h \mid \mathrm{WR}(h) = 1\} = \{(a, b, c) \mid b^2 - 4ac = 0\}$.

Separating polynomial

- Let us try to *characterize* the bivariate degree 2 polynomials $h(x, y)$ such that $\mathsf{WR}(h) = 1$.
- Let $h(x, y) = ax^2 + bxy + cy^2$.
- $X_1 = \{h \mid \mathsf{WR}(h) = 1\} = \{(a, b, c) \mid b^2 - 4ac = 0\}$.
- It helps to **prove** lower bound. For e.g. $\mathsf{WR}(xy) > 1$ because $(0, 1, 0) \notin X_1$.

Separating polynomial

- Let us try to *characterize* the bivariate degree 2 polynomials $h(x, y)$ such that $\mathrm{WR}(h) = 1$.
- Let $h(x, y) = ax^2 + bxy + cy^2$.
- $X_1 = \{h \mid \mathrm{WR}(h) = 1\} = \{(a, b, c) \mid b^2 - 4ac = 0\}$.
- It helps to **prove** lower bound. For e.g. $\mathrm{WR}(xy) > 1$ because $(0, 1, 0) \notin X_1$.
- Such $f = b^2 - 4ac$ is sometimes called a **'polynomial obstruction'** or a **'separating polynomial'**.

Separating polynomial

- Let us try to *characterize* the bivariate degree 2 polynomials $h(x, y)$ such that $\mathrm{WR}(h) = 1$.
- Let $h(x, y) = ax^2 + bxy + cy^2$.
- $X_1 = \{h \mid \mathrm{WR}(h) = 1\} = \{(a, b, c) \mid b^2 - 4ac = 0\}$.
- It helps to **prove** lower bound. For e.g. $\mathrm{WR}(xy) > 1$ because $(0, 1, 0) \notin X_1$.
- Such $f = b^2 - 4ac$ is sometimes called a **'polynomial obstruction'** or a **'separating polynomial'**.
- X_1 is a *closed* set. If there are three sequences (a_n, b_n, c_n) such that $a_n \to a, b_n \to b, c_n \to c$, i.e. limits exist, such that $(a_n, b_n, c_n) \in X_1$, then $(a, b, c) \in X_1$.

Approximation helps

- Example: $\mathsf{WR}(x^2y) \leq 3$, because

Approximation helps

- Example: $\mathsf{WR}(x^2y) \le 3$, because

$$x^2y \;=\; \frac{1}{6}\cdot(x+y)^3 - \frac{1}{6}\cdot(x-y)^3 - \frac{1}{3}\cdot y^3 \;.$$

Approximation helps

- Example: $\mathrm{WR}(x^2y) \le 3$, because

$$x^2y \;=\; \frac{1}{6}\cdot(x+y)^3 - \frac{1}{6}\cdot(x-y)^3 - \frac{1}{3}\cdot y^3\;.$$

- Prove: $\mathrm{WR}(x^2y) = 3$.

Approximation helps

- Example: $\mathsf{WR}(x^2y) \leq 3$, because

$$x^2y \;=\; \frac{1}{6}\cdot(x+y)^3 - \frac{1}{6}\cdot(x-y)^3 - \frac{1}{3}\cdot y^3 \;.$$

- Prove: $\mathsf{WR}(x^2y) = 3$.

- Let $h_\epsilon := \frac{1}{3\epsilon}\left((x+\epsilon y)^3 - x^3\right)$

Approximation helps

- Example: $\mathsf{WR}(x^2y) \leq 3$, because

$$x^2y \;=\; \frac{1}{6}\cdot(x+y)^3 - \frac{1}{6}\cdot(x-y)^3 - \frac{1}{3}\cdot y^3\ .$$

- Prove: $\mathsf{WR}(x^2y) = 3$.

- Let $h_\epsilon \,:=\, \frac{1}{3\epsilon}\left((x+\epsilon y)^3 - x^3\right)$

$$= x^2y + \epsilon xy^2 + \frac{\epsilon^2}{3}y^3 \overset{\epsilon\to 0}{\longrightarrow} x^2y =: h \quad \text{(coefficient-wise).}$$

Approximation helps

- Example: $\mathsf{WR}(x^2y) \leq 3$, because

$$x^2y \;=\; \frac{1}{6}\cdot(x+y)^3 - \frac{1}{6}\cdot(x-y)^3 - \frac{1}{3}\cdot y^3\;.$$

- Prove: $\mathsf{WR}(x^2y) = 3$.

- Let $h_\epsilon := \frac{1}{3\epsilon}\left((x+\epsilon y)^3 - x^3\right)$

$$= x^2y + \epsilon xy^2 + \frac{\epsilon^2}{3}y^3 \overset{\epsilon\to 0}{\longrightarrow} x^2y =: h \quad \text{(coefficient-wise).}$$

- Note: $\mathsf{WR}(h_\epsilon) \leq 2$, for any fixed non-zero ϵ. But $\mathsf{WR}(h) = 3$!

Approximation helps

- Example: $\mathrm{WR}(x^2y) \le 3$, because

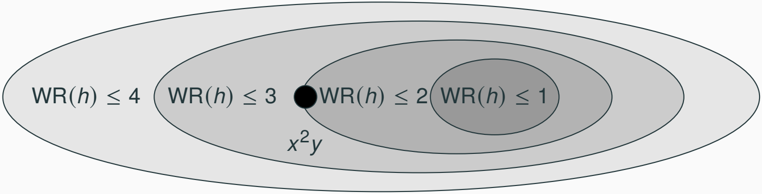
$$x^2y \;=\; \frac{1}{6}\cdot(x+y)^3 - \frac{1}{6}\cdot(x-y)^3 - \frac{1}{3}\cdot y^3\ .$$

- Prove: $\mathrm{WR}(x^2y) = 3$.

- Let $h_\epsilon := \frac{1}{3\epsilon}\left((x+\epsilon y)^3 - x^3\right)$

$$= x^2y + \epsilon xy^2 + \frac{\epsilon^2}{3}y^3 \overset{\epsilon\to 0}{\longrightarrow} x^2y =: h \quad \text{(coefficient-wise).}$$

- Note: $\mathrm{WR}(h_\epsilon) \le 2$, for any fixed non-zero ϵ. But $\mathrm{WR}(h) = 3$!



Border Waring rank

Border Waring rank

- The subtlety: If a **continuous** function (**separating polynomial**) f vanishes on all h with $\mathsf{WR}(h) \leq 2$, then f should also vanishes on x^2y.

Border Waring rank

- The subtlety: If a **continuous** function (**separating polynomial**) f vanishes on all h with $\mathsf{WR}(h) \leq 2$, then f should also vanishes on x^2y.
- To prove $\mathsf{WR}(x^2y) > 2$, we need to find a **discontinuous** function f which vanishes on $\mathsf{WR}(h) \leq 2$ but *does not* vanish on x^2y.

Border Waring rank

- The subtlety: If a **continuous** function (**separating polynomial**) f vanishes on all h with $\mathsf{WR}(h) \leq 2$, then f should also vanishes on x^2y.
- To prove $\mathsf{WR}(x^2y) > 2$, we need to find a **discontinuous** function f which vanishes on $\mathsf{WR}(h) \leq 2$ but *does not* vanish on x^2y.

Border Waring rank

The border Waring rank $\overline{\mathsf{WR}}(h)$ is defined as the smallest n such that h can be **approximated** arbitrarily closely by polynomials of Waring rank $\leq n$.

Border Waring rank

- The subtlety: If a **continuous** function (**separating polynomial**) f vanishes on all h with $\mathsf{WR}(h) \le 2$, then f should also vanishes on x^2y.
- To prove $\mathsf{WR}(x^2y) > 2$, we need to find a **discontinuous** function f which vanishes on $\mathsf{WR}(h) \le 2$ but *does not* vanish on x^2y.

Border Waring rank

The border Waring rank $\overline{\mathsf{WR}}(h)$ is defined as the smallest n such that h can be **approximated** arbitrarily closely by polynomials of Waring rank $\le n$.

- $\overline{\mathsf{WR}}(x^2y) = 2$ but $\mathsf{WR}(x^2y) = 3$.

Border Waring rank

- The subtlety: If a **continuous** function (**separating polynomial**) f vanishes on all h with $\mathsf{WR}(h) \leq 2$, then f should also vanishes on x^2y.
- To prove $\mathsf{WR}(x^2y) > 2$, we need to find a **discontinuous** function f which vanishes on $\mathsf{WR}(h) \leq 2$ but *does not* vanish on x^2y.

Border Waring rank

The border Waring rank $\overline{\mathsf{WR}}(h)$ is defined as the smallest n such that h can be **approximated** arbitrarily closely by polynomials of Waring rank $\leq n$.

- $\overline{\mathsf{WR}}(x^2y) = 2$ but $\mathsf{WR}(x^2y) = 3$.
- The subtlety is *gone*: $X_n := \{h \mid \overline{\mathsf{WR}}(h) \leq n\}$, is now a **closed** set.

Border Waring rank

- The subtlety: If a **continuous** function (**separating polynomial**) f vanishes on all h with $\mathsf{WR}(h) \leq 2$, then f should also vanishes on x^2y.
- To prove $\mathsf{WR}(x^2y) > 2$, we need to find a **discontinuous** function f which vanishes on $\mathsf{WR}(h) \leq 2$ but *does not* vanish on x^2y.

Border Waring rank

The border Waring rank $\overline{\mathsf{WR}}(h)$ is defined as the smallest n such that h can be **approximated** arbitrarily closely by polynomials of Waring rank $\leq n$.

- $\overline{\mathsf{WR}}(x^2y) = 2$ but $\mathsf{WR}(x^2y) = 3$.
- The subtlety is *gone*: $X_n := \{h \mid \overline{\mathsf{WR}}(h) \leq n\}$, is now a **closed** set.
- On to proving lower bounds: To show $\overline{\mathsf{WR}}(p) > n$, for some p, it suffices to show that $p \notin X_n$, i.e. find a *continuous* function f that vanishes on X_n but not on p.

Border complexity

- Replace Waring rank by any sensible measure Γ. It can be size, dc, pc and so on.

Border complexity

- Replace Waring rank by any sensible measure Γ. It can be size, dc, pc and so on.
- For any Γ, we can define the border complexity measure $\overline{\Gamma}$ via:
  $\overline{\Gamma}(h)$ is the *smallest* n such that $h(\mathbf{x})$ can be approximated arbitrarily closely by polynomials $h_\epsilon(\mathbf{x})$ with $\Gamma(h_\epsilon) \leq n$.

Border complexity

- Replace Waring rank by any sensible measure Γ. It can be $\mathsf{size}, \mathsf{dc}, \mathsf{pc}$ and so on.
- For any Γ, we can define the border complexity measure $\overline{\Gamma}$ via:
  $\overline{\Gamma}(h)$ is the *smallest* n such that $h(\boldsymbol{x})$ can be approximated arbitrarily closely by polynomials $h_\epsilon(\boldsymbol{x})$ with $\Gamma(h_\epsilon) \leq n$. In other words,

$$\lim_{\epsilon \to 0} h_\epsilon = h \text{ (coefficient-wise)} .$$

Border complexity

- Replace Waring rank by any sensible measure Γ. It can be size, dc, pc and so on.
- For any Γ, we can define the border complexity measure $\overline{\Gamma}$ via:
  $\overline{\Gamma}(h)$ is the *smallest* n such that $h(\boldsymbol{x})$ can be approximated arbitrarily closely by polynomials $h_\epsilon(\boldsymbol{x})$ with $\Gamma(h_\epsilon) \le n$. In other words,

$$\lim_{\epsilon \to 0} h_\epsilon = h \text{ (coefficient-wise)} .$$

- Important border rank: **border tensor rank**, related to border Waring rank!

Border complexity

- Replace Waring rank by any sensible measure Γ. It can be size, dc, pc and so on.
- For any Γ, we can define the border complexity measure $\overline{\Gamma}$ via:
  $\overline{\Gamma}(h)$ is the *smallest* n such that $h(\boldsymbol{x})$ can be approximated arbitrarily closely by polynomials $h_\epsilon(\boldsymbol{x})$ with $\Gamma(h_\epsilon) \leq n$. In other words,

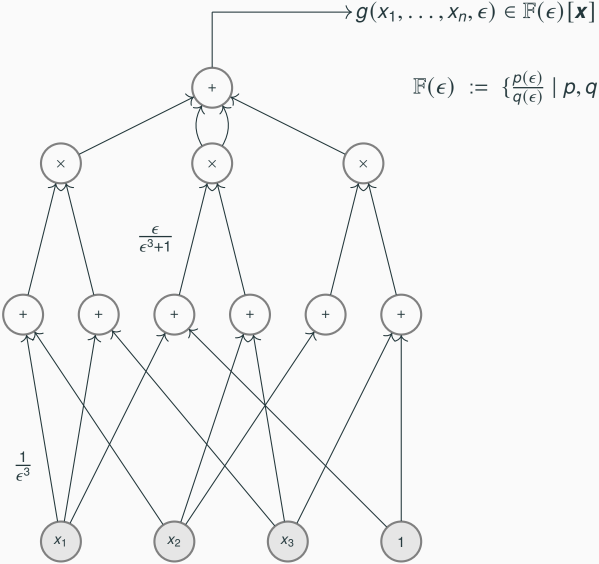
$$\lim_{\epsilon \to 0} h_\epsilon = h \text{ (coefficient-wise)} .$$

- Important border rank: **border tensor rank**, related to border Waring rank!
  Border tensor rank is *directly* related to the matrix multiplication exponent ω [Bini 1980, Coppersmith-Winograd 1990].

Approximative circuits

- Coefficients in the earlier definition can be arbitrary depending on the parameter ϵ. Can it be 'nicer'?

- Coefficients in the earlier definition can be arbitrary depending on the parameter ϵ. Can it be 'nicer'?
- Yes! Via *'approximative circuits'*.



$$\mathbb{F}(\epsilon) := \{\tfrac{p(\epsilon)}{q(\epsilon)} \mid p, q \in \mathbb{F}[\epsilon], q(\epsilon) \neq 0\}$$

Algebraic approximation

Algebraic approximation

- Suppose, we assume the following:
  - $g(\boldsymbol{x}, \epsilon) \in \mathbb{F}[x_1, \ldots, x_n, \epsilon]$, i.e. it is a polynomial of the form

$$g(\boldsymbol{x}, \epsilon) = \sum_{i=0}^{M} g_i(x_1, \ldots, x_n) \cdot \epsilon^i ,$$

Algebraic approximation

- Suppose, we assume the following:
  - $g(\boldsymbol{x}, \epsilon) \in \mathbb{F}[x_1, \ldots, x_n, \epsilon]$, i.e. it is a polynomial of the form

$$g(\boldsymbol{x}, \epsilon) = \sum_{i=0}^{M} g_i(x_1, \ldots, x_n) \cdot \epsilon^i ,$$

  - Can we say anything about the complexity of g_0?

Algebraic approximation

- Suppose, we assume the following:
  - $g(\boldsymbol{x}, \epsilon) \in \mathbb{F}[x_1, \ldots, x_n, \epsilon]$, i.e. it is a polynomial of the form

$$g(\boldsymbol{x}, \epsilon) \;=\; \sum_{i=0}^{M} g_i(x_1, \ldots, x_n) \cdot \epsilon^i \;,$$

  - Can we say anything about the complexity of g_0?
- Obvious attempt:
  - Since, $g(\boldsymbol{x}, 0) = g_0$, why not just set $\epsilon = 0$?!

Algebraic approximation

- Suppose, we assume the following:
  - $g(\boldsymbol{x}, \epsilon) \in \mathbb{F}[x_1, \ldots, x_n, \epsilon]$, i.e. it is a polynomial of the form

$$g(\boldsymbol{x}, \epsilon) \;=\; \sum_{i=0}^{M} g_i(x_1, \ldots, x_n) \cdot \epsilon^i \;,$$

  - Can we say anything about the complexity of g_0?
- Obvious attempt:
  - Since, $g(\boldsymbol{x}, 0) = g_0$, why not just set $\epsilon = 0$?! Setting $\epsilon = 0$ *may not* be 'legal' as it could be using $1/\epsilon$ in the wire. Though it is well-defined!

Algebraic approximation

- Suppose, we assume the following:
  - $g(\boldsymbol{x}, \epsilon) \in \mathbb{F}[x_1, \ldots, x_n, \epsilon]$, i.e. it is a polynomial of the form

$$g(\boldsymbol{x}, \epsilon) \;=\; \sum_{i=0}^{M} g_i(x_1, \ldots, x_n) \cdot \epsilon^i \;,$$

  - Can we say anything about the complexity of g_0?
- Obvious attempt:
  - Since, $g(\boldsymbol{x}, 0) = g_0$, why not just set $\epsilon = 0$?! Setting $\epsilon = 0$ *may not* be 'legal' as it could be using $1/\epsilon$ in the wire. Though it is well-defined!
- Summary: g_0 is really something **non-trivial** and being 'approximated' by the circuit since $\lim_{\epsilon \to 0} g(\boldsymbol{x}, \epsilon) = g_0$.

Algebraic approximation (continued)

Algebraic Approximation [Bürgisser 2004]

A polynomial $h(x) \in \mathbb{F}[\boldsymbol{x}]$ approximative complexity s, if there is a $g(\boldsymbol{x}, \epsilon) \in \mathbb{F}(\epsilon)[\boldsymbol{x}]$, of size s, over $\mathbb{F}(\epsilon)$, and a polynomial $S(\boldsymbol{x}, \epsilon) \in \mathbb{F}[\epsilon][\boldsymbol{x}]$ such that $g(\boldsymbol{x}, \epsilon) = h(\boldsymbol{x}) + \epsilon \cdot S(\boldsymbol{x}, \epsilon)$. In other words, $\lim_{\epsilon \to 0} g = h$.

Algebraic Approximation [Bürgisser 2004]

A polynomial $h(x) \in \mathbb{F}[\boldsymbol{x}]$ approximative complexity s, if there is a $g(\boldsymbol{x}, \epsilon) \in \mathbb{F}(\epsilon)[\boldsymbol{x}]$, of size s, over $\mathbb{F}(\epsilon)$, and a polynomial $S(\boldsymbol{x}, \epsilon) \in \mathbb{F}[\epsilon][\boldsymbol{x}]$ such that $g(\boldsymbol{x}, \epsilon) = h(\boldsymbol{x}) + \epsilon \cdot S(\boldsymbol{x}, \epsilon)$. In other words, $\lim_{\epsilon \to 0} g = h$.

- $\overline{\text{size}}(h) \leq \text{size}(h)$.

Algebraic approximation (continued)

Algebraic Approximation [Bürgisser 2004]

A polynomial $h(x) \in \mathbb{F}[\boldsymbol{x}]$ approximative complexity s, if there is a $g(\boldsymbol{x}, \epsilon) \in \mathbb{F}(\epsilon)[\boldsymbol{x}]$, of size s, over $\mathbb{F}(\epsilon)$, and a polynomial $S(\boldsymbol{x}, \epsilon) \in \mathbb{F}[\epsilon][\boldsymbol{x}]$ such that $g(\boldsymbol{x}, \epsilon) = h(\boldsymbol{x}) + \epsilon \cdot S(\boldsymbol{x}, \epsilon)$. In other words, $\lim_{\epsilon \to 0} g = h$.

- $\overline{\text{size}}(h) \leq \text{size}(h)$.
- If g has circuit of size s over $\mathbb{F}(\epsilon)$, then one can assume that the highest degree of ϵ in g can be *exponentially large* 2^{s^2} [Bürgisser 2004, 2020].

Algebraic approximation (continued)

Algebraic Approximation [Bürgisser 2004]

A polynomial $h(x) \in \mathbb{F}[\boldsymbol{x}]$ approximative complexity s, if there is a $g(\boldsymbol{x}, \epsilon) \in \mathbb{F}(\epsilon)[\boldsymbol{x}]$, of size s, over $\mathbb{F}(\epsilon)$, and a polynomial $S(\boldsymbol{x}, \epsilon) \in \mathbb{F}[\epsilon][\boldsymbol{x}]$ such that $g(\boldsymbol{x}, \epsilon) = h(\boldsymbol{x}) + \epsilon \cdot S(\boldsymbol{x}, \epsilon)$. In other words, $\lim_{\epsilon \to 0} g = h$.

- $\overline{\text{size}}(h) \leq \text{size}(h)$.
- If g has circuit of size s over $\mathbb{F}(\epsilon)$, then one can assume that the highest degree of ϵ in g can be *exponentially large* 2^{s^2} [Bürgisser 2004, 2020].
- Let us assume that $g(\boldsymbol{x}, \epsilon) = \sum_{i=0}^{M} g_i \epsilon^i$, where $M = 2^{s^2}$. Note: $g_0 = h$.

Algebraic Approximation [Bürgisser 2004]

A polynomial $h(x) \in \mathbb{F}[\boldsymbol{x}]$ approximative complexity s, if there is a $g(\boldsymbol{x}, \epsilon) \in \mathbb{F}(\epsilon)[\boldsymbol{x}]$, of size s, over $\mathbb{F}(\epsilon)$, and a polynomial $S(\boldsymbol{x}, \epsilon) \in \mathbb{F}[\epsilon][\boldsymbol{x}]$ such that $g(\boldsymbol{x}, \epsilon) = h(\boldsymbol{x}) + \epsilon \cdot S(\boldsymbol{x}, \epsilon)$. In other words, $\lim_{\epsilon \to 0} g = h$.

- $\overline{\text{size}}(h) \leq \text{size}(h)$.
- If g has circuit of size s over $\mathbb{F}(\epsilon)$, then one can assume that the highest degree of ϵ in g can be *exponentially large* 2^{s^2} [Bürgisser 2004, 2020].
- Let us assume that $g(\boldsymbol{x}, \epsilon) = \sum_{i=0}^{M} g_i \epsilon^i$, where $M = 2^{s^2}$. Note: $g_0 = h$.
    - Pick $M + 1$ many distinct values from $\mathbb{F}$ *randomly* and interpolate;

Algebraic approximation (continued)

Algebraic Approximation [Bürgisser 2004]

A polynomial $h(x) \in \mathbb{F}[\boldsymbol{x}]$ approximative complexity s, if there is a $g(\boldsymbol{x}, \epsilon) \in \mathbb{F}(\epsilon)[\boldsymbol{x}]$, of size s, over $\mathbb{F}(\epsilon)$, and a polynomial $S(\boldsymbol{x}, \epsilon) \in \mathbb{F}[\epsilon][\boldsymbol{x}]$ such that $g(\boldsymbol{x}, \epsilon) = h(\boldsymbol{x}) + \epsilon \cdot S(\boldsymbol{x}, \epsilon)$. In other words, $\lim_{\epsilon \to 0} g = h$.

- $\overline{\text{size}}(h) \leq \text{size}(h)$.
- If g has circuit of size s over $\mathbb{F}(\epsilon)$, then one can assume that the highest degree of ϵ in g can be *exponentially large* 2^{s^2} [Bürgisser 2004, 2020].
- Let us assume that $g(\boldsymbol{x}, \epsilon) = \sum_{i=0}^{M} g_i \epsilon^i$, where $M = 2^{s^2}$. Note: $g_0 = h$.
  - Pick $M + 1$ many distinct values from $\mathbb{F}$ *randomly* and interpolate;
  - $\text{size}(h) \leq \exp(\overline{\text{size}}(h))$.

Algebraic approximation (continued)

Algebraic Approximation [Bürgisser 2004]

A polynomial $h(x) \in \mathbb{F}[\boldsymbol{x}]$ approximative complexity s, if there is a $g(\boldsymbol{x}, \epsilon) \in \mathbb{F}(\epsilon)[\boldsymbol{x}]$, of size s, over $\mathbb{F}(\epsilon)$, and a polynomial $S(\boldsymbol{x}, \epsilon) \in \mathbb{F}[\epsilon][\boldsymbol{x}]$ such that $g(\boldsymbol{x}, \epsilon) = h(\boldsymbol{x}) + \epsilon \cdot S(\boldsymbol{x}, \epsilon)$. In other words, $\lim_{\epsilon \to 0} g = h$.

- $\overline{\text{size}}(h) \leq \text{size}(h)$.
- If g has circuit of size s over $\mathbb{F}(\epsilon)$, then one can assume that the highest degree of ϵ in g can be *exponentially large* 2^{s^2} [Bürgisser 2004, 2020].
- Let us assume that $g(\boldsymbol{x}, \epsilon) = \sum_{i=0}^{M} g_i \epsilon^i$, where $M = 2^{s^2}$. Note: $g_0 = h$.
  - Pick $M + 1$ many distinct values from $\mathbb{F}$ *randomly* and interpolate;
  - $\text{size}(h) \leq \exp(\overline{\text{size}}(h))$.
- $\overline{\text{size}}(h) \leq \text{size}(h) \leq \exp(\overline{\text{size}}(h))$

De-bordering

- De-bordering: Given a ‘nice’ class $\mathcal{C}$, can we de-border $\overline{\mathcal{C}}$? i.e. find another ’nice’ class $\mathcal{D}$ such that $\overline{\mathcal{C}} \subseteq \mathcal{D}$?

De-bordering

- De-bordering: Given a ‘nice’ class $\mathcal{C}$, can we de-border $\overline{\mathcal{C}}$? i.e. find another ’nice’ class $\mathcal{D}$ such that $\overline{\mathcal{C}} \subseteq \mathcal{D}$?

- Take $\mathcal{C} \in \{\mathsf{VBP}, \mathsf{VP}, \Sigma \wedge \Sigma, \mathsf{VNP}, \cdots\}$.

De-bordering

- De-bordering: Given a 'nice' class $\mathcal{C}$, can we de-border $\overline{\mathcal{C}}$? i.e. find another 'nice' class $\mathcal{D}$ such that $\overline{\mathcal{C}} \subseteq \mathcal{D}$?

- Take $\mathcal{C} \in \{\mathsf{VBP}, \mathsf{VP}, \Sigma \wedge \Sigma, \mathsf{VNP}, \cdots\}$.

- Major open questions from [Mulmuley Sohoni 2001] and [Bürgisser 2001]:

De-bordering

- De-bordering: Given a 'nice' class $\mathcal{C}$, can we de-border $\overline{\mathcal{C}}$? i.e. find another 'nice' class $\mathcal{D}$ such that $\overline{\mathcal{C}} \subseteq \mathcal{D}$?
- Take $\mathcal{C} \in \{\mathsf{VBP}, \mathsf{VP}, \Sigma \wedge \Sigma, \mathsf{VNP}, \cdots\}$.
- Major open questions from [Mulmuley Sohoni 2001] and [Bürgisser 2001]:

$$\overline{\mathsf{VBP}} \stackrel{?}{=} \mathsf{VBP}\ ,\ \overline{\mathsf{VP}} \stackrel{?}{=} \mathsf{VP}\ ,\ \overline{\mathsf{VNP}} \stackrel{?}{=} \mathsf{VNP}\ .$$

Strengthening lower bounds and its advantages

Strengthening lower bounds and its advantages

Strengthening Valiant's Conjecture [Milind Sohoni 2001]

$\mathsf{VNP} \not\subset \overline{\mathsf{VBP}}$ & $\mathsf{VNP} \not\subset \overline{\mathsf{VP}}$. Equivalently, $\overline{\mathsf{dc}}(\mathsf{perm}_n)$ and $\overline{\mathrm{size}}(\mathsf{perm}_n)$ are both $n^{\omega(1)}$.

Strengthening lower bounds and its advantages

Strengthening Valiant's Conjecture [Milind Sohoni 2001]

$\mathsf{VNP} \not\subset \overline{\mathsf{VBP}}$ & $\mathsf{VNP} \not\subset \overline{\mathsf{VP}}$. Equivalently, $\overline{\mathsf{dc}}(\mathsf{perm}_n)$ and $\overline{\text{size}}(\mathsf{perm}_n)$ are both $n^{\omega(1)}$.

- Both det and perm have 'nice' **symmetries**.

Strengthening lower bounds and its advantages

Strengthening Valiant's Conjecture [Milind Sohoni 2001]

$\mathsf{VNP} \not\subset \overline{\mathsf{VBP}}$ & $\mathsf{VNP} \not\subset \overline{\mathsf{VP}}$. Equivalently, $\overline{\mathsf{dc}}(\mathsf{perm}_n)$ and $\overline{\mathsf{size}}(\mathsf{perm}_n)$ are both $n^{\omega(1)}$.

- Both det and perm have 'nice' **symmetries**.
- Symmetry-characterization **avoids** the Razborov–Rudich barrier: *Very few* functions are symmetry-characterized, so symmetry-characterization violates the largeness criterion!

De-bordering results and their importance

De-bordering results and their importance

- A few known de-bordering results:

De-bordering results and their importance

- A few known de-bordering results:
  - $\overline{\mathsf{VBP}_{\text{non-com}}} = \mathsf{VBP}_{\text{non-com}}$, in the noncommutative world [Nisan 1991].

De-bordering results and their importance

- A few known de-bordering results:
  - $\overline{\mathsf{VBP}_{\text{non-com}}} = \mathsf{VBP}_{\text{non-com}}$, in the noncommutative world [Nisan 1991].
  - $\overline{\Sigma \wedge \Sigma} \subsetneq \mathsf{VBP}$ [Forbes 2016, Bläser-Dörfler-Ikenmeyer 2021].

De-bordering results and their importance

- A few known de-bordering results:
  - $\overline{\mathsf{VBP}_{\text{non-com}}} = \mathsf{VBP}_{\text{non-com}}$, in the noncommutative world [Nisan 1991].
  - $\overline{\Sigma \wedge \Sigma} \subsetneq \mathsf{VBP}$ [Forbes 2016, Bläser-Dörfler-Ikenmeyer 2021].
  - $\overline{\Sigma^{[s]}\Pi} = \Sigma^{[s]}\Pi$ and $\overline{\Pi^{[d]}\Sigma} = \Pi^{[d]}\Sigma$.

De-bordering results and their importance

- A few known de-bordering results:
  - $\overline{\mathsf{VBP}_{\text{non-com}}} = \mathsf{VBP}_{\text{non-com}}$, in the noncommutative world [Nisan 1991].
  - $\overline{\Sigma \wedge \Sigma} \subsetneq \mathsf{VBP}$ [Forbes 2016, Bläser-Dörfler-Ikenmeyer 2021].
  - $\overline{\Sigma^{[s]}\Pi} = \Sigma^{[s]}\Pi$ and $\overline{\Pi^{[d]}\Sigma} = \Pi^{[d]}\Sigma$.
- Upper bounds and lower bounds are *dual* to each other.

De-bordering results and their importance

- A few known de-bordering results:
  - $\overline{\mathsf{VBP}_{\text{non-com}}} = \mathsf{VBP}_{\text{non-com}}$, in the noncommutative world [Nisan 1991].
  - $\overline{\Sigma \wedge \Sigma} \subsetneq \mathsf{VBP}$ [Forbes 2016, Bläser-Dörfler-Ikenmeyer 2021].
  - $\overline{\Sigma^{[s]}\Pi} = \Sigma^{[s]}\Pi$ and $\overline{\Pi^{[d]}\Sigma} = \Pi^{[d]}\Sigma$.
- Upper bounds and lower bounds are *dual* to each other.
- Further potential applications in identity testing and understanding its 'robustness'.

Border depth-3 circuits

Depth-3 circuits

Depth-3 circuits

- Depth-3 circuits with top fan-in k, are denoted as $\Sigma^{[k]}\Pi^{[d]}\Sigma$. Thus, the size is trivially bounded by $O(knd)$.

Depth-3 circuits

- Depth-3 circuits with top fan-in k, are denoted as $\Sigma^{[k]}\Pi^{[d]}\Sigma$. Thus, the size is trivially bounded by $O(knd)$.
- They compute polynomials of the form $\sum_{i=1}^{k}\prod_{j=1}^{d} \ell_{ij}$, where ℓ_{ij} are linear polynomials (i.e. $a_0 + a_1x_1 + \ldots + a_nx_n$, for $a_i \in \mathbb{F}$).

Depth-3 circuits

- Depth-3 circuits with top fan-in k, are denoted as $\Sigma^{[k]}\Pi^{[d]}\Sigma$. Thus, the size is trivially bounded by $O(knd)$.
- They compute polynomials of the form $\sum_{i=1}^{k}\prod_{j=1}^{d}\ell_{ij}$, where ℓ_{ij} are linear polynomials (i.e. $a_0 + a_1x_1 + \ldots + a_nx_n$, for $a_i \in \mathbb{F}$).
- How powerful are $\Sigma^{[k]}\Pi^{[d]}\Sigma$ circuits? Are they *universal*?

Depth-3 circuits

- Depth-3 circuits with top fan-in k, are denoted as $\Sigma^{[k]}\Pi^{[d]}\Sigma$. Thus, the size is trivially bounded by $O(knd)$.
- They compute polynomials of the form $\sum_{i=1}^{k}\prod_{j=1}^{d}\ell_{ij}$, where ℓ_{ij} are linear polynomials (i.e. $a_0 + a_1x_1 + \ldots + a_nx_n$, for $a_i \in \mathbb{F}$).
- How powerful are $\Sigma^{[k]}\Pi^{[d]}\Sigma$ circuits? Are they *universal*?
- **No**.

Depth-3 circuits

- Depth-3 circuits with top fan-in k, are denoted as $\Sigma^{[k]}\Pi^{[d]}\Sigma$. Thus, the size is trivially bounded by $O(knd)$.
- They compute polynomials of the form $\sum_{i=1}^{k}\prod_{j=1}^{d}\ell_{ij}$, where ℓ_{ij} are linear polynomials (i.e. $a_0 + a_1x_1 + \ldots + a_nx_n$, for $a_i \in \mathbb{F}$).
- How powerful are $\Sigma^{[k]}\Pi^{[d]}\Sigma$ circuits? Are they *universal*?
- **No**. E.g. the *Inner Product* polynomial $\langle \boldsymbol{x}, \boldsymbol{y}\rangle = x_1y_1 + \ldots + x_{k+1}y_{k+1}$ **cannot** be written as a $\Sigma^{[k]}\Pi^{[d]}\Sigma$ circuit, *regardless* of the product fan-in d!

Power of border depth-3 circuits

- What about $\overline{\Sigma^{[k]}\Pi^{[d]}\Sigma}$ circuits?

Power of border depth-3 circuits

- What about $\overline{\Sigma^{[k]}\Pi^{[d]}\Sigma}$ circuits?
- Recall: $h \in \overline{\Sigma^{[k]}\Pi^{[d]}\Sigma}$ of size s if there exists a polynomial g such that

Power of border depth-3 circuits

- What about $\overline{\Sigma^{[k]}\Pi^{[d]}\Sigma}$ circuits?
- Recall: $h \in \overline{\Sigma^{[k]}\Pi^{[d]}\Sigma}$ of size s if there exists a polynomial g such that

$$g(\boldsymbol{x}, \epsilon) = h(\boldsymbol{x}) + \epsilon \cdot S(\boldsymbol{x}, \epsilon) \ ,$$

Power of border depth-3 circuits

- What about $\overline{\Sigma^{[k]}\Pi^{[d]}\Sigma}$ circuits?
- Recall: $h \in \overline{\Sigma^{[k]}\Pi^{[d]}\Sigma}$ of size s if there exists a polynomial g such that

$$g(\boldsymbol{x}, \epsilon) = h(\boldsymbol{x}) + \epsilon \cdot S(\boldsymbol{x}, \epsilon) \ ,$$

where g can be computed by a $\Sigma^{[k]}\Pi^{[d]}\Sigma$ circuit, over $\mathbb{F}(\epsilon)$, of size s.

Power of border depth-3 circuits

- What about $\overline{\Sigma^{[k]}\Pi^{[d]}\Sigma}$ circuits?
- Recall: $h \in \overline{\Sigma^{[k]}\Pi^{[d]}\Sigma}$ of size s if there exists a polynomial g such that

$$g(\boldsymbol{x}, \epsilon) = h(\boldsymbol{x}) + \epsilon \cdot S(\boldsymbol{x}, \epsilon) \ ,$$

where g can be computed by a $\Sigma^{[k]}\Pi^{[d]}\Sigma$ circuit, over $\mathbb{F}(\epsilon)$, of size s.

Border depth-3 fan-in 2 circuits are 'universal' [Kumar 2020]

Let P be *any* homogeneous n-variate degree d polynomial. Then, $P \in \overline{\Sigma^{[2]}\Pi^{[D]}\Sigma}$, where $D := \exp(n, d)$.

De-bordering $\overline{\Sigma^{[2]}\Pi\Sigma}$ circuits

De-bordering $\overline{\Sigma^{[2]}\Pi\Sigma}$ circuits

- If h is approximated by a $\Sigma^{[2]}\Pi^{[d]}\Sigma$ circuit with $d = \mathrm{poly}(n)$, what's the *exact* complexity of h?

De-bordering $\overline{\Sigma^{[2]}\Pi\Sigma}$ circuits

- If h is approximated by a $\Sigma^{[2]}\Pi^{[d]}\Sigma$ circuit with $d = \mathrm{poly}(n)$, what's the *exact* complexity of h?
  - Is it even explicit?

De-bordering $\overline{\Sigma^{[2]}\Pi\Sigma}$ circuits

- If h is approximated by a $\Sigma^{[2]}\Pi^{[d]}\Sigma$ circuit with $d = \mathrm{poly}(n)$, what's the *exact* complexity of h?
  - Is it even explicit? If yes, $\overline{\Sigma^{[2]}\Pi^{[d]}\Sigma} \subseteq \mathsf{VNP}$?

De-bordering $\overline{\Sigma^{[2]}\Pi\Sigma}$ circuits

- If h is approximated by a $\Sigma^{[2]}\Pi^{[d]}\Sigma$ circuit with $d = \mathsf{poly}(n)$, what's the *exact* complexity of h?
  - Is it even explicit? If yes, $\overline{\Sigma^{[2]}\Pi^{[d]}\Sigma} \subseteq \mathsf{VNP}$?

Theorem 1 (Border of polynomial-sized depth-3 top-fanin-2 circuits are 'easy') [Dutta-Dwivedi-Saxena FOCS 2021].

$\overline{\Sigma^{[2]}\Pi^{[d]}\Sigma} \subseteq \mathsf{VBP}$, for $d = \mathsf{poly}(n)$. In particular, any polynomial in the border of top-fanin-2 size-s depth-3 circuits, can also be exactly computed by a linear projection of a $\mathsf{poly}(s) \times \mathsf{poly}(s)$ determinant.

De-bordering $\overline{\Sigma^{[2]}\Pi\Sigma}$ circuits

- If h is approximated by a $\Sigma^{[2]}\Pi^{[d]}\Sigma$ circuit with $d = \mathsf{poly}(n)$, what's the *exact* complexity of h?
  - Is it even explicit? If yes, $\overline{\Sigma^{[2]}\Pi^{[d]}\Sigma} \subseteq \mathsf{VNP}$?

Theorem 1 (Border of polynomial-sized depth-3 top-fanin-2 circuits are 'easy') [Dutta-Dwivedi-Saxena FOCS 2021].

$\overline{\Sigma^{[2]}\Pi^{[d]}\Sigma} \subseteq \mathsf{VBP}$, for $d = \mathsf{poly}(n)$. In particular, any polynomial in the border of top-fanin-2 size-s depth-3 circuits, can also be exactly computed by a linear projection of a $\mathsf{poly}(s) \times \mathsf{poly}(s)$ determinant.

Remark. The result holds if one replaces the top-fanin-2 by arbitrary constant k.

Why $k = 2$ is hard to analyze?

Why $k = 2$ is hard to analyze?

- Non-trivial cancellations for $k = 2$ make things harder.

Why $k = 2$ is hard to analyze?

- Non-trivial cancellations for $k = 2$ make things harder.
- E.g., $T_1 := \epsilon^{-3}(1+\epsilon x_1+\epsilon^2 x_2+\epsilon^3 x_3+\ldots)$, $T_2 := \epsilon^{-3}(1+\epsilon x_1+\epsilon^2 x_2+\epsilon^3 x_4+\ldots)$.

Why $k = 2$ is hard to analyze?

- Non-trivial cancellations for $k = 2$ make things harder.
- E.g., $T_1 := \epsilon^{-3}(1+\epsilon x_1+\epsilon^2 x_2+\epsilon^3 x_3+\ldots)$, $T_2 := \epsilon^{-3}(1+\epsilon x_1+\epsilon^2 x_2+\epsilon^3 x_4+\ldots)$. Note, $\lim_{\epsilon\to 0} (T_1 - T_2) = (x_3 - x_4)$.

Why $k = 2$ is hard to analyze?

- Non-trivial cancellations for $k = 2$ make things harder.
- E.g., $T_1 := \epsilon^{-3}(1+\epsilon x_1+\epsilon^2 x_2+\epsilon^3 x_3+\ldots)$, $T_2 := \epsilon^{-3}(1+\epsilon x_1+\epsilon^2 x_2+\epsilon^3 x_4+\ldots)$. Note, $\lim_{\epsilon\to 0}\ (T_1 - T_2) \ = \ (x_3 - x_4)$.
- Note $x^2 \equiv (x - \epsilon^{M/2}\cdot a)(x+\epsilon^{M/2}\cdot a) \mod \epsilon^M$, for *any* $a \in \mathbb{F}$.

Why $k = 2$ is hard to analyze?

- Non-trivial cancellations for $k = 2$ make things harder.
- E.g., $T_1 := \epsilon^{-3}(1+\epsilon x_1+\epsilon^2 x_2+\epsilon^3 x_3+\ldots)$, $T_2 := \epsilon^{-3}(1+\epsilon x_1+\epsilon^2 x_2+\epsilon^3 x_4+\ldots)$. Note, $\lim_{\epsilon\to 0} (T_1 - T_2) = (x_3 - x_4)$.
- Note $x^2 \equiv (x - \epsilon^{M/2}\cdot a)(x + \epsilon^{M/2}\cdot a) \mod \epsilon^M$, for *any* $a \in \mathbb{F}$.
- Moreover,

$$\lim_{\epsilon\to 0} \frac{1}{\epsilon^M}\cdot\left(x^2 - (x - \epsilon^{M/2}\cdot a)(x + \epsilon^{M/2}\cdot a)\right) = a^2 .$$

Why $k = 2$ is hard to analyze?

- Non-trivial cancellations for $k = 2$ make things harder.
- E.g., $T_1 := \epsilon^{-3}(1+\epsilon x_1+\epsilon^2 x_2+\epsilon^3 x_3+\ldots)$, $T_2 := \epsilon^{-3}(1+\epsilon x_1+\epsilon^2 x_2+\epsilon^3 x_4+\ldots)$.
  Note, $\lim_{\epsilon\to 0}\,(T_1 - T_2) \;=\; (x_3 - x_4)$.
- Note $x^2 \equiv (x - \epsilon^{M/2}\cdot a)(x + \epsilon^{M/2}\cdot a) \mod \epsilon^M$, for *any* $a \in \mathbb{F}$.
- Moreover,

$$\lim_{\epsilon\to 0}\frac{1}{\epsilon^M}\cdot\left(x^2 \;-\; (x-\epsilon^{M/2}\cdot a)(x+\epsilon^{M/2}\cdot a)\right) \;=\; a^2\ .$$

- *Infinitely* many factorizations may give *infinitely* many limits.

Proof sketch for $k = 2$

Proof sketch for $k = 2$

- $T_1 + T_2 = f(\boldsymbol{x}) + \epsilon \cdot S(\boldsymbol{x}, \epsilon)$, where $T_i \in \Pi\Sigma \in \mathbb{F}(\epsilon)[\boldsymbol{x}]$. Assume $\deg(f) = d$.

Proof sketch for $k = 2$

- $T_1 + T_2 = f(\boldsymbol{x}) + \epsilon \cdot S(\boldsymbol{x}, \epsilon)$, where $T_i \in \Pi\Sigma \in \mathbb{F}(\epsilon)[\boldsymbol{x}]$. Assume $\deg(f) = d$.
- Very broad idea: reduce fanin 2 to 1 with a 'nice' form.

Proof sketch for $k = 2$

- $T_1 + T_2 = f(\boldsymbol{x}) + \epsilon \cdot S(\boldsymbol{x}, \epsilon)$, where $T_i \in \Pi\Sigma \in \mathbb{F}(\epsilon)[\boldsymbol{x}]$. Assume $\mathsf{deg}(f) = d$.
- Very broad idea: reduce fanin 2 to 1 with a 'nice' form.
- Apply a map Φ, defined by $\Phi : x_i \mapsto z \cdot x_i + \alpha_i$, where $\alpha_i \in \mathbb{F}$ are *random*.

Proof sketch for $k = 2$

- $T_1 + T_2 = f(\boldsymbol{x}) + \epsilon \cdot S(\boldsymbol{x}, \epsilon)$, where $T_i \in \Pi\Sigma \in \mathbb{F}(\epsilon)[\boldsymbol{x}]$. Assume $\deg(f) = d$.
- Very broad idea: reduce fanin 2 to 1 with a 'nice' form.
- Apply a map Φ, defined by $\Phi : x_i \mapsto z \cdot x_i + \alpha_i$, where $\alpha_i \in \mathbb{F}$ are *random*.
  - The variable z is the "degree counter",

Proof sketch for $k = 2$

- $T_1 + T_2 = f(\boldsymbol{x}) + \epsilon \cdot S(\boldsymbol{x}, \epsilon)$, where $T_i \in \Pi\Sigma \in \mathbb{F}(\epsilon)[\boldsymbol{x}]$. Assume $\deg(f) = d$.
- Very broad idea: reduce fanin 2 to 1 with a 'nice' form.
- Apply a map Φ, defined by $\Phi : x_i \mapsto z \cdot x_i + \alpha_i$, where $\alpha_i \in \mathbb{F}$ are *random*.
  - The variable z is the "degree counter",
  - α_i to make sure: If $\ell \mid T_i$, then $\Phi(\ell)|_{z=0} = \ell(\alpha_1, \ldots, \alpha_n) \in \mathbb{F}(\epsilon)\setminus\{0\}$.

Proof sketch for $k = 2$

- $T_1 + T_2 = f(\boldsymbol{x}) + \epsilon \cdot S(\boldsymbol{x}, \epsilon)$, where $T_i \in \Pi\Sigma \in \mathbb{F}(\epsilon)[\boldsymbol{x}]$. Assume $\deg(f) = d$.
- Very broad idea: reduce fanin 2 to 1 with a 'nice' form.
- Apply a map Φ, defined by $\Phi : x_i \mapsto z \cdot x_i + \alpha_i$, where $\alpha_i \in \mathbb{F}$ are *random*.
  - The variable z is the "degree counter",
  - α_i to make sure: If $\ell \mid T_i$, then $\Phi(\ell)|_{z=0} = \ell(\alpha_1, \ldots, \alpha_n) \in \mathbb{F}(\epsilon)\backslash\{0\}$.
- It suffices to show that $\Phi(f)$ has small ABP.

Proof sketch for $k = 2$

- $T_1 + T_2 = f(\boldsymbol{x}) + \epsilon \cdot S(\boldsymbol{x}, \epsilon)$, where $T_i \in \Pi\Sigma \in \mathbb{F}(\epsilon)[\boldsymbol{x}]$. Assume $\mathsf{deg}(f) = d$.
- Very broad idea: reduce fanin 2 to 1 with a 'nice' form.
- Apply a map Φ, defined by $\Phi : x_i \mapsto z \cdot x_i + \alpha_i$, where $\alpha_i \in \mathbb{F}$ are *random*.
  - The variable z is the "degree counter",
  - α_i to make sure: If $\ell \mid T_i$, then $\Phi(\ell)|_{z=0} = \ell(\alpha_1, \ldots, \alpha_n) \in \mathbb{F}(\epsilon) \backslash \{0\}$.
- It suffices to show that $\Phi(f)$ has small ABP.
- We devise a technique called DiDIL - **Di**vide, **D**erive, **I**nterpolate with **L**imit.

$k = 2$ proof continued: Divide and Derive

$k = 2$ proof continued: Divide and Derive

- Let $\Phi(T_i) =: \epsilon^{a_i} \cdot \tilde{T}_i$, for $i \in [2]$, where $a_i := \mathsf{val}_\epsilon\,(\Phi(T_i))$.
- $\mathsf{val}_\epsilon\,(\cdot)$ denotes the highest power of ϵ dividing it.

$k = 2$ proof continued: Divide and Derive

- Let $\Phi(T_i) =: \epsilon^{a_i} \cdot \tilde{T}_i$, for $i \in [2]$, where $a_i := \mathsf{val}_\epsilon\,(\Phi(T_i))$.
- $\mathsf{val}_\epsilon\,(\cdot)$ denotes the highest power of ϵ dividing it.
- **Di**vide and **D**erive:

$k = 2$ proof continued: Divide and Derive

- Let $\Phi(T_i) =: \epsilon^{a_i} \cdot \tilde{T}_i$, for $i \in [2]$, where $a_i := \mathsf{val}_\epsilon\,(\Phi(T_i))$.
- $\mathsf{val}_\epsilon\,(\cdot)$ denotes the highest power of ϵ dividing it.
- **Di**vide and **De**rive:

$$
\begin{aligned}
f + \epsilon \cdot S =&\ T_1 + T_2 \\
\implies \Phi(f) + \epsilon \cdot \Phi(S) =&\ \Phi(T_1) + \Phi(T_2) \\
\implies \Phi(f)/\tilde{T}_2 + \epsilon \cdot \Phi(S)/\tilde{T}_2 =&\ \epsilon^{a_2} + \Phi(T_1)/\tilde{T}_2 \\
\implies \partial_z\left(\Phi(f)/\tilde{T}_2\right) + \epsilon \cdot \partial_z\left(\Phi(S)/\tilde{T}_2\right) =&\ \partial_z\left(\Phi(T_1)/\tilde{T}_2\right) =: g_1\ . \qquad (1)
\end{aligned}
$$

$k = 2$ proof continued: Divide and Derive

- Let $\Phi(T_i) =: \epsilon^{a_i} \cdot \tilde{T}_i$, for $i \in [2]$, where $a_i := \mathrm{val}_\epsilon\,(\Phi(T_i))$.
- $\mathrm{val}_\epsilon\,(\cdot)$ denotes the highest power of ϵ dividing it.
- **Di**vide and **De**rive:

$$\begin{aligned} f + \epsilon \cdot S &= T_1 + T_2 \\ \implies \Phi(f) + \epsilon \cdot \Phi(S) &= \Phi(T_1) + \Phi(T_2) \\ \implies \Phi(f)/\tilde{T}_2 + \epsilon \cdot \Phi(S)/\tilde{T}_2 &= \epsilon^{a_2} + \Phi(T_1)/\tilde{T}_2 \\ \implies \partial_z\left(\Phi(f)/\tilde{T}_2\right) + \epsilon \cdot \partial_z\left(\Phi(S)/\tilde{T}_2\right) &= \partial_z\left(\Phi(T_1)/\tilde{T}_2\right) =: g_1 \,. \end{aligned} \tag{1}$$

- $\lim_{\epsilon\to 0} g_1 = 1/t_2 \cdot \partial_z(\Phi(f))$, where $t_2 := \lim_{\epsilon\to 0} \tilde{T}_2$.

$k = 2$ proof continued

$k = 2$ proof continued

- First target: compute $\lim_{\epsilon \to 0} g_1 = 1/t_2 \cdot \partial_z(\Phi(f))$.

$k = 2$ proof continued

- First target: compute $\lim_{\epsilon \to 0} g_1 = 1/t_2 \cdot \partial_z(\Phi(f))$.
- Logarithmic derivative: $\mathrm{dlog}_z(h) := \partial_z(h)/h$.

$k = 2$ proof continued

- First target: compute $\lim_{\epsilon \to 0} g_1 = 1/t_2 \cdot \partial_z(\Phi(f))$.
- Logarithmic derivative: $\mathrm{dlog}_z(h) := \partial_z(h)/h$.
- dlog *linearizes* product: $\mathrm{dlog}(h_1 h_2) = \mathrm{dlog}(h_1) + \mathrm{dlog}(h_2)$.

$k = 2$ proof continued

- First target: compute $\lim_{\epsilon \to 0} g_1 = 1/t_2 \cdot \partial_z(\Phi(f))$.
- Logarithmic derivative: $\mathrm{dlog}_z(h) := \partial_z(h)/h$.
- dlog *linearizes* product: $\mathrm{dlog}(h_1 h_2) = \mathrm{dlog}(h_1) + \mathrm{dlog}(h_2)$. Note:

$$\begin{aligned}
\partial_z\left(\Phi(T_1)/\tilde{T}_2\right) &= \Phi(T_1)/\tilde{T}_2 \cdot \mathrm{dlog}\left(\Phi(T_1)/\tilde{T}_2\right) \\
&= (\Pi\Sigma/\Pi\Sigma) \cdot \mathrm{dlog}\,(\Pi\Sigma/\Pi\Sigma) \\
&= \Pi\Sigma/\Pi\Sigma \cdot \left(\pm \sum \mathrm{dlog}(\Sigma)\right).
\end{aligned}$$

$k = 2$ proof continued

- First target: compute $\lim_{\epsilon \to 0} g_1 = 1/t_2 \cdot \partial_z(\Phi(f))$.
- Logarithmic derivative: $\mathsf{dlog}_z(h) := \partial_z(h)/h$.
- dlog *linearizes* product: $\mathsf{dlog}(h_1 h_2) = \mathsf{dlog}(h_1) + \mathsf{dlog}(h_2)$. Note:

$$\begin{aligned} \partial_z\left(\Phi(T_1)/\tilde{T}_2\right) &= \Phi(T_1)/\tilde{T}_2 \cdot \mathsf{dlog}\left(\Phi(T_1)/\tilde{T}_2\right) \\ &= (\Pi\Sigma/\Pi\Sigma) \cdot \mathsf{dlog}\,(\Pi\Sigma/\Pi\Sigma) \\ &= \Pi\Sigma/\Pi\Sigma \cdot \left(\pm \sum \mathsf{dlog}(\Sigma)\right). \end{aligned}$$

- Here Σ means just a linear polynomial ℓ.

$k = 2$ proof continued: Quick recap

$k = 2$ proof continued: Quick recap

- Recap: $1/t_2 \cdot \partial_z(\Phi(f)) \; = \; \lim_{\epsilon \to 0} g_1 \; = \; \lim_{\epsilon \to 0} \; (\Pi\Sigma/\Pi\Sigma) \cdot (\pm \sum \mathrm{dlog}(\Sigma))$.

$k = 2$ proof continued: Quick recap

- Recap: $1/t_2 \cdot \partial_z(\Phi(f)) \; = \; \lim_{\epsilon \to 0} g_1 \; = \; \lim_{\epsilon \to 0} \; (\Pi\Sigma/\Pi\Sigma) \cdot (\pm \sum \mathsf{dlog}(\Sigma))$.

- $\deg(f) = d \implies \deg_z(\Phi(f)) = d \implies \deg_z(\partial_z(\Phi(f))) = d - 1$.

$k = 2$ proof continued: Quick recap

- Recap: $1/t_2 \cdot \partial_z(\Phi(f)) \; = \; \lim_{\epsilon \to 0} g_1 \; = \; \lim_{\epsilon \to 0} \; (\Pi\Sigma/\Pi\Sigma) \cdot (\pm \sum \mathrm{dlog}(\Sigma))$.
- $\deg(f) = d \implies \deg_z(\Phi(f)) = d \implies \deg_z(\partial_z(\Phi(f))) = d - 1$.
- Suffices to compute $\lim_{\epsilon \to 0} g_1 \mod z^d$.

$k = 2$ **proof:** dlog **strikes!**

$k = 2$ proof: dlog strikes!

- What is $\mathsf{dlog}(\ell)$ for a linear polynomial $\ell = A - z \cdot B$?

$k = 2$ proof: dlog strikes!

- What is $\mathrm{dlog}(\ell)$ for a linear polynomial $\ell = A - z \cdot B$?

$$\begin{aligned}\mathrm{dlog}(A - zB) &= -\frac{B}{A\left(1 - z \cdot B/A\right)} \\ &= -\frac{B}{A} \cdot \sum_{j=0}^{d-1} \left(\frac{z \cdot B}{A}\right)^j \mod z^d \\ &\in \Sigma \wedge \Sigma \,.\end{aligned}$$

$k = 2$ proof: dlog strikes!

- What is $\mathsf{dlog}(\ell)$ for a linear polynomial $\ell = A - z \cdot B$?

$$\begin{aligned}
\mathsf{dlog}(A - zB) &= -\frac{B}{A\,(1 - z \cdot B/A)} \\
&= -\frac{B}{A} \cdot \sum_{j=0}^{d-1} \left(\frac{z \cdot B}{A}\right)^j \mod z^d \\
&\in \Sigma \wedge \Sigma\,.
\end{aligned}$$

Thus,

$$\begin{aligned}
\lim_{\epsilon \to 0} g_1 \mod z^d &\equiv \lim_{\epsilon \to 0} \Pi\Sigma/\Pi\Sigma \cdot \left(\sum \mathsf{dlog}(\Sigma)\right) \mod z^d \\
&\equiv \lim_{\epsilon \to 0} (\Pi\Sigma/\Pi\Sigma) \cdot (\Sigma \wedge \Sigma) \mod z^d \\
&\in \overline{(\Pi\Sigma/\Pi\Sigma) \cdot (\Sigma \wedge \Sigma)} \mod z^d\,.
\end{aligned}$$

Finishing the proof

Finishing the proof

- $\overline{\mathcal{C} \cdot \mathcal{D}} \subseteq \overline{\mathcal{C}} \cdot \overline{\mathcal{D}}$. Therefore,

Finishing the proof

- $\overline{\mathcal{C} \cdot \mathcal{D}} \subseteq \overline{\mathcal{C}} \cdot \overline{\mathcal{D}}$. Therefore,

$$\begin{aligned} \overline{(\Pi\Sigma/\Pi\Sigma) \cdot (\Sigma \wedge \Sigma)} &\subseteq \overline{(\Pi\Sigma/\Pi\Sigma)} \cdot \overline{\Sigma \wedge \Sigma} \\ &\subseteq (\mathsf{ABP}/\mathsf{ABP}) \cdot \mathsf{ABP} \\ &= \mathsf{ABP}/\mathsf{ABP}\ . \end{aligned}$$

Finishing the proof

- $\overline{\mathcal{C} \cdot \mathcal{D}} \subseteq \overline{\mathcal{C}} \cdot \overline{\mathcal{D}}$. Therefore,

$$\begin{aligned} \overline{(\Pi\Sigma/\Pi\Sigma) \cdot (\Sigma \wedge \Sigma)} &\subseteq \overline{(\Pi\Sigma/\Pi\Sigma)} \cdot \overline{\Sigma \wedge \Sigma} \\ &\subseteq (\mathsf{ABP}/\mathsf{ABP}) \cdot \mathsf{ABP} \\ &= \mathsf{ABP}/\mathsf{ABP}\ . \end{aligned}$$

- Eliminate division to get: $\lim_{\epsilon \to 0} g_1 \mod z^d \equiv \mathsf{ABP}/\mathsf{ABP} \mod z^d = \mathsf{ABP}$.

Finishing the proof

- $\overline{\mathcal{C} \cdot \mathcal{D}} \subseteq \overline{\mathcal{C}} \cdot \overline{\mathcal{D}}$. Therefore,

$$\begin{aligned}\overline{(\Pi\Sigma/\Pi\Sigma)\cdot(\Sigma\wedge\Sigma)} &\subseteq \overline{(\Pi\Sigma/\Pi\Sigma)} \cdot \overline{\Sigma\wedge\Sigma} \\ &\subseteq (\mathsf{ABP}/\mathsf{ABP})\cdot\mathsf{ABP} \\ &= \mathsf{ABP}/\mathsf{ABP}\ .\end{aligned}$$

- Eliminate division to get: $\lim_{\epsilon\to 0} g_1 \mod z^d \equiv \mathsf{ABP}/\mathsf{ABP} \mod z^d = \mathsf{ABP}$.
- Thus, $1/t_2 \cdot \partial_z(\Phi(f)) = \lim_{\epsilon\to 0} g_1 = \mathsf{ABP}$.

Finishing the proof

- $\overline{\mathcal{C} \cdot \mathcal{D}} \subseteq \overline{\mathcal{C}} \cdot \overline{\mathcal{D}}$. Therefore,

$$\begin{aligned} \overline{(\Pi\Sigma/\Pi\Sigma) \cdot (\Sigma \wedge \Sigma)} &\subseteq \overline{(\Pi\Sigma/\Pi\Sigma)} \cdot \overline{\Sigma \wedge \Sigma} \\ &\subseteq (\mathsf{ABP}/\mathsf{ABP}) \cdot \mathsf{ABP} \\ &= \mathsf{ABP}/\mathsf{ABP} \;. \end{aligned}$$

- Eliminate division to get: $\lim_{\epsilon \to 0} g_1 \mod z^d \equiv \mathsf{ABP}/\mathsf{ABP} \mod z^d = \mathsf{ABP}$.
- Thus, $1/t_2 \cdot \partial_z(\Phi(f)) = \lim_{\epsilon \to 0} g_1 = \mathsf{ABP}$.
- Thus, $\Phi(f)/t_2 = \mathsf{ABP} \implies \Phi(f) = \mathsf{ABP} \implies f = \mathsf{ABP}$.

Derandomizing border depth-3 circuits

Polynomial Identity Testing

Polynomial Identity Testing

- **Polynomial Identity Testing (PIT):** Given a circuit C, test whether C computes the *zero* polynomial *(deterministically)*.

Polynomial Identity Testing

- **Polynomial Identity Testing (PIT):** Given a circuit C, test whether C computes the *zero* polynomial *(deterministically)*.
  - *Blackbox*-PIT asks for an algorithm to test the zeroness of a given algebraic circuit via mere *query access*.

Polynomial Identity Testing

- **Polynomial Identity Testing (PIT):** Given a circuit C, test whether C computes the *zero* polynomial *(deterministically)*.
  - *Blackbox*-PIT asks for an algorithm to test the zeroness of a given algebraic circuit via mere *query access*.

Polynomial Identity Lemma [Ore, Demillo-Lipton, Schwartz, Zippel]

Polynomial Identity Testing

- **Polynomial Identity Testing (PIT):** Given a circuit C, test whether C computes the *zero* polynomial *(deterministically)*.
  - *Blackbox*-PIT asks for an algorithm to test the zeroness of a given algebraic circuit via mere *query access*.

Polynomial Identity Lemma [Ore, Demillo-Lipton, Schwartz, Zippel]

If $P(\boldsymbol{x})$ is a nonzero polynomial of degree d, and $S \subseteq \mathbb{F}$ of size at least $d+1$, then $P(\boldsymbol{a}) \neq 0$ for some $\boldsymbol{a} \in S^n$.

Polynomial Identity Testing

- **Polynomial Identity Testing (PIT):** Given a circuit C, test whether C computes the *zero* polynomial *(deterministically)*.
  - *Blackbox*-PIT asks for an algorithm to test the zeroness of a given algebraic circuit via mere *query access*.

Polynomial Identity Lemma [Ore, Demillo-Lipton, Schwartz, Zippel]

If $P(\mathbf{x})$ is a nonzero polynomial of degree d, and $S \subseteq \mathbb{F}$ of size at least $d+1$, then $P(\mathbf{a}) \neq 0$ for some $\mathbf{a} \in S^n$.

- This above lemma puts PIT $\in \mathsf{RP}$.

Polynomial Identity Testing

- **Polynomial Identity Testing (PIT):** Given a circuit C, test whether C computes the *zero* polynomial *(deterministically)*.
  - *Blackbox*-PIT asks for an algorithm to test the zeroness of a given algebraic circuit via mere *query access*.

Polynomial Identity Lemma [Ore, Demillo-Lipton, Schwartz, Zippel]

If $P(\boldsymbol{x})$ is a nonzero polynomial of degree d, and $S \subseteq \mathbb{F}$ of size at least $d+1$, then $P(\boldsymbol{a}) \neq 0$ for some $\boldsymbol{a} \in S^n$.

- This above lemma puts PIT $\in \mathsf{RP}$.
- Can we *derandomize* blackbox-PIT?

Polynomial Identity Testing

- **Polynomial Identity Testing (PIT):** Given a circuit C, test whether C computes the *zero* polynomial *(deterministically)*.
  - *Blackbox*-PIT asks for an algorithm to test the zeroness of a given algebraic circuit via mere *query access*.

Polynomial Identity Lemma [Ore, Demillo-Lipton, Schwartz, Zippel]

If $P(\boldsymbol{x})$ is a nonzero polynomial of degree d, and $S \subseteq \mathbb{F}$ of size at least $d+1$, then $P(\boldsymbol{a}) \neq 0$ for some $\boldsymbol{a} \in S^n$.

- This above lemma puts PIT $\in \mathsf{RP}$.
- Can we *derandomize* blackbox-PIT? Some special cases are derandomized.

Polynomial Identity Testing

- **Polynomial Identity Testing (PIT):** Given a circuit C, test whether C computes the *zero* polynomial *(deterministically)*.
  - *Blackbox*-PIT asks for an algorithm to test the zeroness of a given algebraic circuit via mere *query access*.

Polynomial Identity Lemma [Ore, Demillo-Lipton, Schwartz, Zippel]

If $P(\boldsymbol{x})$ is a nonzero polynomial of degree d, and $S \subseteq \mathbb{F}$ of size at least $d+1$, then $P(\boldsymbol{a}) \neq 0$ for some $\boldsymbol{a} \in S^n$.

- This above lemma puts PIT $\in \mathsf{RP}$.
- Can we *derandomize* blackbox-PIT? Some special cases are derandomized.
- Derandomizing PIT, for restricted cases, has many algorithmic applications:

Polynomial Identity Testing

- **Polynomial Identity Testing (PIT):** Given a circuit C, test whether C computes the *zero* polynomial *(deterministically)*.
  - *Blackbox*-PIT asks for an algorithm to test the zeroness of a given algebraic circuit via mere *query access*.

Polynomial Identity Lemma [Ore, Demillo-Lipton, Schwartz, Zippel]

If $P(\boldsymbol{x})$ is a nonzero polynomial of degree d, and $S \subseteq \mathbb{F}$ of size at least $d+1$, then $P(\boldsymbol{a}) \neq 0$ for some $\boldsymbol{a} \in S^n$.

- This above lemma puts PIT $\in \mathsf{RP}$.
- Can we *derandomize* blackbox-PIT? Some special cases are derandomized.
- Derandomizing PIT, for restricted cases, has many algorithmic applications:
  - Graph Theory [Lovasz'79], [Fenner-Gurjar-Theirauf'19]

Polynomial Identity Testing

- **Polynomial Identity Testing (PIT):** Given a circuit C, test whether C computes the *zero* polynomial *(deterministically)*.
  - *Blackbox*-PIT asks for an algorithm to test the zeroness of a given algebraic circuit via mere *query access*.

Polynomial Identity Lemma [Ore, Demillo-Lipton, Schwartz, Zippel]

If $P(\boldsymbol{x})$ is a nonzero polynomial of degree d, and $S \subseteq \mathbb{F}$ of size at least $d+1$, then $P(\boldsymbol{a}) \neq 0$ for some $\boldsymbol{a} \in S^n$.

- This above lemma puts PIT $\in \mathsf{RP}$.
- Can we *derandomize* blackbox-PIT? Some special cases are derandomized.
- Derandomizing PIT, for restricted cases, has many algorithmic applications:
  - Graph Theory [Lovasz'79], [Fenner-Gurjar-Theirauf'19]
  - Primality Testing [Agrawal-Kayal-Saxena'04].

Border PIT

Border PIT

Border hitting set

$\mathcal{H}$ is a hitting set for a class $\overline{\mathcal{C}}$, if $g(x, \epsilon) \in \mathcal{C}_{\mathbb{F}(\epsilon)}$ approximates a non-zero polynomial $h(\boldsymbol{x})$, then $\exists \boldsymbol{a} \in \mathcal{H}$ such that $g(\boldsymbol{a}, \epsilon) \notin \epsilon \cdot \mathbb{F}[\epsilon]$, i.e. $h(\boldsymbol{a}) \neq 0$.

Border PIT

Border hitting set

$\mathcal{H}$ is a hitting set for a class $\overline{\mathcal{C}}$, if $g(x, \epsilon) \in \mathcal{C}_{\mathbb{F}(\epsilon)}$ approximates a non-zero polynomial $h(\boldsymbol{x})$, then $\exists \boldsymbol{a} \in \mathcal{H}$ such that $g(\boldsymbol{a}, \epsilon) \notin \epsilon \cdot \mathbb{F}[\epsilon]$, i.e. $h(\boldsymbol{a}) \neq 0$.

- Finding $a \in \mathbb{F}^n$ such that $g(\boldsymbol{a}, \epsilon) \neq 0$ *does not* suffice.

Border PIT

Border hitting set

$\mathcal{H}$ is a hitting set for a class $\overline{\mathcal{C}}$, if $g(x,\epsilon) \in \mathcal{C}_{\mathbb{F}(\epsilon)}$ approximates a non-zero polynomial $h(\boldsymbol{x})$, then $\exists \boldsymbol{a} \in \mathcal{H}$ such that $g(\boldsymbol{a},\epsilon) \notin \epsilon \cdot \mathbb{F}[\epsilon]$, i.e. $h(\boldsymbol{a}) \neq 0$.

- Finding $a \in \mathbb{F}^n$ such that $g(\boldsymbol{a},\epsilon) \neq 0$ *does not* suffice.
- h could have really high complexity compared to g.

Border PIT

Border hitting set

$\mathcal{H}$ is a hitting set for a class $\overline{\mathcal{C}}$, if $g(x, \epsilon) \in \mathcal{C}_{\mathbb{F}(\epsilon)}$ approximates a non-zero polynomial $h(\boldsymbol{x})$, then $\exists \boldsymbol{a} \in \mathcal{H}$ such that $g(\boldsymbol{a}, \epsilon) \notin \epsilon \cdot \mathbb{F}[\epsilon]$, i.e. $h(\boldsymbol{a}) \neq 0$.

- Finding $a \in \mathbb{F}^n$ such that $g(\boldsymbol{a}, \epsilon) \neq 0$ *does not* suffice.
- h could have really high complexity compared to g.
- We know

Border PIT

Border hitting set

$\mathcal{H}$ is a hitting set for a class $\overline{C}$, if $g(x, \epsilon) \in C_{\mathbb{F}(\epsilon)}$ approximates a non-zero polynomial $h(\boldsymbol{x})$, then $\exists \boldsymbol{a} \in \mathcal{H}$ such that $g(\boldsymbol{a}, \epsilon) \notin \epsilon \cdot \mathbb{F}[\epsilon]$, i.e. $h(\boldsymbol{a}) \neq 0$.

- Finding $a \in \mathbb{F}^n$ such that $g(\boldsymbol{a}, \epsilon) \neq 0$ *does not* suffice.
- h could have really high complexity compared to g.
- We know
  - polynomial-time hitting set for $\overline{\Pi\Sigma} = \Pi\Sigma$ [Klivans-Spielman 2001],

Border PIT

Border hitting set

$\mathcal{H}$ is a hitting set for a class $\overline{\mathcal{C}}$, if $g(x,\epsilon) \in \mathcal{C}_{\mathbb{F}(\epsilon)}$ approximates a non-zero polynomial $h(\boldsymbol{x})$, then $\exists \boldsymbol{a} \in \mathcal{H}$ such that $g(\boldsymbol{a},\epsilon) \notin \epsilon \cdot \mathbb{F}[\epsilon]$, i.e. $h(\boldsymbol{a}) \neq 0$.

- Finding $a \in \mathbb{F}^n$ such that $g(\boldsymbol{a},\epsilon) \neq 0$ *does not* suffice.
- h could have really high complexity compared to g.
- We know
  - polynomial-time hitting set for $\overline{\Pi\Sigma} = \Pi\Sigma$ [Klivans-Spielman 2001],
  - *quasi*polynomial-time hitting set for $\overline{\Sigma \wedge \Sigma}$ [Forbes-Shpilka 20013].

Border PIT

Border hitting set

$\mathcal{H}$ is a hitting set for a class $\overline{\mathcal{C}}$, if $g(x,\epsilon) \in \mathcal{C}_{\mathbb{F}(\epsilon)}$ approximates a non-zero polynomial $h(\boldsymbol{x})$, then $\exists \boldsymbol{a} \in \mathcal{H}$ such that $g(\boldsymbol{a},\epsilon) \notin \epsilon \cdot \mathbb{F}[\epsilon]$, i.e. $h(\boldsymbol{a}) \neq 0$.

- Finding $a \in \mathbb{F}^n$ such that $g(\boldsymbol{a},\epsilon) \neq 0$ *does not* suffice.
- h could have really high complexity compared to g.
- We know
  - polynomial-time hitting set for $\overline{\Pi\Sigma} = \Pi\Sigma$ [Klivans-Spielman 2001],
  - *quasi*polynomial-time hitting set for $\overline{\Sigma \wedge \Sigma}$ [Forbes-Shpilka 20013].
- $n^{O(k)}$-time hitting set is known for $\Sigma^{[k]}\Pi\Sigma$ [Saxena-Seshadri 2012].

Border hitting set

$\mathcal{H}$ is a hitting set for a class $\overline{\mathcal{C}}$, if $g(x,\epsilon) \in \mathcal{C}_{\mathbb{F}(\epsilon)}$ approximates a non-zero polynomial $h(\boldsymbol{x})$, then $\exists \boldsymbol{a} \in \mathcal{H}$ such that $g(\boldsymbol{a},\epsilon) \notin \epsilon \cdot \mathbb{F}[\epsilon]$, i.e. $h(\boldsymbol{a}) \neq 0$.

- Finding $a \in \mathbb{F}^n$ such that $g(\boldsymbol{a},\epsilon) \neq 0$ *does not* suffice.
- h could have really high complexity compared to g.
- We know
  - polynomial-time hitting set for $\overline{\Pi\Sigma} = \Pi\Sigma$ [Klivans-Spielman 2001],
  - *quasi*polynomial-time hitting set for $\overline{\Sigma \wedge \Sigma}$ [Forbes-Shpilka 20013].
- $n^{O(k)}$-time hitting set is known for $\underline{\Sigma^{[k]}\Pi\Sigma}$ [Saxena-Seshadri 2012]. Unfortunately, it *does not* work for $\overline{\Sigma^{[k]}\Pi\Sigma}$.

Border PIT

Border hitting set

$\mathcal{H}$ is a hitting set for a class $\overline{\mathcal{C}}$, if $g(x,\epsilon) \in \mathcal{C}_{\mathbb{F}(\epsilon)}$ approximates a non-zero polynomial $h(\boldsymbol{x})$, then $\exists \boldsymbol{a} \in \mathcal{H}$ such that $g(\boldsymbol{a},\epsilon) \notin \epsilon \cdot \mathbb{F}[\epsilon]$, i.e. $h(\boldsymbol{a}) \neq 0$.

- Finding $a \in \mathbb{F}^n$ such that $g(\boldsymbol{a},\epsilon) \neq 0$ *does not* suffice.
- h could have really high complexity compared to g.
- We know
  - polynomial-time hitting set for $\overline{\Pi\Sigma} = \Pi\Sigma$ [Klivans-Spielman 2001],
  - *quasi*polynomial-time hitting set for $\overline{\Sigma \wedge \Sigma}$ [Forbes-Shpilka 20013].
- $n^{O(k)}$-time hitting set is known for $\underline{\Sigma^{[k]}\Pi\Sigma}$ [Saxena-Seshadri 2012]. Unfortunately, it *does not* work for $\overline{\Sigma^{[k]}\Pi\Sigma}$.
- General PIT for det is not known!

Border PIT

Border hitting set

$\mathcal{H}$ is a hitting set for a class $\overline{\mathcal{C}}$, if $g(x, \epsilon) \in \mathcal{C}_{\mathbb{F}(\epsilon)}$ approximates a non-zero polynomial $h(\boldsymbol{x})$, then $\exists \boldsymbol{a} \in \mathcal{H}$ such that $g(\boldsymbol{a}, \epsilon) \notin \epsilon \cdot \mathbb{F}[\epsilon]$, i.e. $h(\boldsymbol{a}) \neq 0$.

- Finding $a \in \mathbb{F}^n$ such that $g(\boldsymbol{a}, \epsilon) \neq 0$ *does not* suffice.
- h could have really high complexity compared to g.
- We know
  - polynomial-time hitting set for $\overline{\Pi\Sigma} = \Pi\Sigma$ [Klivans-Spielman 2001],
  - *quasi*polynomial-time hitting set for $\overline{\Sigma \wedge \Sigma}$ [Forbes-Shpilka 20013].
- $n^{O(k)}$-time hitting set is known for $\underline{\Sigma^{[k]}\Pi\Sigma}$ [Saxena-Seshadri 2012]. Unfortunately, it *does not* work for $\overline{\Sigma^{[k]}\Pi\Sigma}$.
- General PIT for det is not known!

Theorem 2 (Derandomizing polynomal-sized depth-3 top-fanin-*k* circuits) [Dutta-Dwivedi-Saxena 2021]

There exists an explicit quasipolynomial-time ($s^{O(\log\log s)}$) hitting set for size-s $\overline{\Sigma^{[k]}\Pi\Sigma}$ circuits, for any constant k.

Conclusion

Concluding remarks

Concluding remarks

- Can we show $\overline{\Sigma^{[k]}\Pi^{[d]}\Sigma} \neq \mathsf{VBP}$, for $d = \mathsf{poly}(n)$?

Concluding remarks

- Can we show $\overline{\Sigma^{[k]}\Pi^{[d]}\Sigma} \neq \mathsf{VBP}$, for $d = \mathsf{poly}(n)$?
  - Fix any constant $k \geq 1$. There is an explicit n-variate and $< n$ degree polynomial f such that f can be computed by a $\overline{\Sigma^{[k+1]}\Pi^{[d]}\Sigma}$ circuit of size $O(n)$ such that f requires $2^{\Omega(n)}$-size $\overline{\Sigma^{[k]}\Pi^{[D]}\Sigma}$ circuits. [Dutta-Saxena 2021, Preprint]

Concluding remarks

- Can we show $\overline{\Sigma^{[k]}\Pi^{[d]}\Sigma} \neq \mathsf{VBP}$, for $d = \mathsf{poly}(n)$?
  - Fix any constant $k \geq 1$. There is an explicit n-variate and $< n$ degree polynomial f such that f can be computed by a $\overline{\Sigma^{[k+1]}\Pi^{[d]}\Sigma}$ circuit of size $O(n)$ such that f requires $2^{\Omega(n)}$-size $\overline{\Sigma^{[k]}\Pi^{[D]}\Sigma}$ circuits. [Dutta-Saxena 2021, Preprint]
  - Thus, the above result also shows $\overline{\Sigma^{[k]}\Pi^{[d]}\Sigma} \neq \mathsf{VBP}, \mathsf{VNP}$.

Concluding remarks

- Can we show $\overline{\Sigma^{[k]}\Pi^{[d]}\Sigma} \neq \mathsf{VBP}$, for $d = \mathsf{poly}(n)$?
  - Fix any constant $k \geq 1$. There is an explicit n-variate and $< n$ degree polynomial f such that f can be computed by a $\Sigma^{[k+1]}\Pi^{[d]}\Sigma$ circuit of size $O(n)$ such that f requires $2^{\Omega(n)}$-size $\overline{\Sigma^{[k]}\Pi^{[D]}\Sigma}$ circuits. [Dutta-Saxena 2021, Preprint]
  - Thus, the above result also shows $\overline{\Sigma^{[k]}\Pi^{[d]}\Sigma} \neq \mathsf{VBP}, \mathsf{VNP}$.

- [Dutta-Dwivedi-Saxena 2021] showed a *quasi*polynomial-time hitting set for $\overline{\Sigma^{[k]}\Pi^{[d]}\Sigma}$ circuits. Can we improve it to polynomial?

Concluding remarks

- Can we show $\overline{\Sigma^{[k]}\Pi^{[d]}\Sigma} \neq \mathsf{VBP}$, for $d = \mathrm{poly}(n)$?
  - Fix any constant $k \geq 1$. There is an explicit n-variate and $< n$ degree polynomial f such that f can be computed by a $\overline{\Sigma^{[k+1]}\Pi^{[d]}\Sigma}$ circuit of size $O(n)$ such that f requires $2^{\Omega(n)}$-size $\overline{\Sigma^{[k]}\Pi^{[D]}\Sigma}$ circuits. [Dutta-Saxena 2021, Preprint]
  - Thus, the above result also shows $\overline{\Sigma^{[k]}\Pi^{[d]}\Sigma} \neq \mathsf{VBP}, \mathsf{VNP}$.
- [Dutta-Dwivedi-Saxena 2021] showed a *quasi*polynomial-time hitting set for $\overline{\Sigma^{[k]}\Pi^{[d]}\Sigma}$ circuits. Can we improve it to polynomial?
- Does our technique extend to arbitrary constant-depth border circuits? Currently it extends to very restricted depth-4 and depth-5 circuits.

Concluding remarks

- Can we show $\overline{\Sigma^{[k]}\Pi^{[d]}\Sigma} \neq \mathsf{VBP}$, for $d = \mathsf{poly}(n)$?
  - Fix any constant $k \geq 1$. There is an explicit n-variate and $< n$ degree polynomial f such that f can be computed by a $\Sigma^{[k+1]}\Pi^{[d]}\Sigma$ circuit of size $O(n)$ such that f requires $2^{\Omega(n)}$-size $\overline{\Sigma^{[k]}\Pi^{[D]}\Sigma}$ circuits. [Dutta-Saxena 2021, Preprint]
  - Thus, the above result also shows $\overline{\Sigma^{[k]}\Pi^{[d]}\Sigma} \neq \mathsf{VBP}, \mathsf{VNP}$.

- [Dutta-Dwivedi-Saxena 2021] showed a *quasi*polynomial-time hitting set for $\overline{\Sigma^{[k]}\Pi^{[d]}\Sigma}$ circuits. Can we improve it to polynomial?

- Does our technique extend to arbitrary constant-depth border circuits? Currently it extends to very restricted depth-4 and depth-5 circuits.

Thank you & stay safe!