

# Algebraic Circuit Complexity

A journey

Sankalp Mittal

Undergraduate Project

February 3, 2024

# Outline

## 1 Arithmetic Circuits

- Background
- Different Models
  - Circuits
  - Formula
  - ABP

## 2 Sum of ABP

- smABP
- ROABP
- Hardness Bootstrapping
  - Steps Towards Proof

## 3 Border Complexity

- Presentable is Explicit
  - Interpolation
  - Valiant's Criterion

## 4 References

# Some History...

The “Holy Grail” of Complexity Theory is the  $P \stackrel{?}{\neq} NP$  problem. This problem has been unexpectedly hard to solve.

Leslie Valiant hypothesised  $VP \stackrel{?}{\neq} VNP$  problem as a stepping stone towards the  $P \stackrel{?}{\neq} NP$  problem.

Recent works in Complexity Theory have been trying to show the separation between various classes that have sprung out of Valiant’s initial classes.

# Why do we need Circuits?

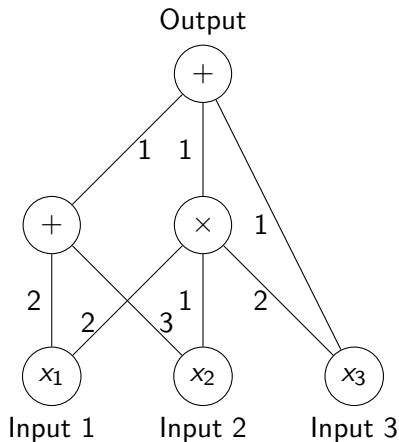
“What is the best way to compute a given polynomial  $f(x_1, \dots, x_n)$  from basic operations such as  $+$  and  $\times$ ?” This is the main motivating problem in the field of arithmetic circuit complexity. The notion of complexity of a polynomial is measured via the size of the smallest arithmetic circuit computing it. Arithmetic circuits provide a robust model of computation for polynomials. [Sap15]

# Algebraic Circuit

## Circuit

An **Algebraic Circuit** is formally a **Directed Acyclic Graph** with a unique *sink vertex* called the *root* such that each internal vertex is labelled as  $+$  or  $\times$ .

## Example



In this example, the final output will be  $2x_1 + 3x_2 + 2x_1x_2 + x_3$

# Complexity Classes

There are classes corresponding the **P** and **NP** called **Valiant's P (VP)** and **Valiant's NP (VNP)**.

- **VP**: Consists of polynomials whose size and degree are both polynomially bounded.
- **VNP**: Set of all polynomials  $f \in \mathbb{F}[x_1, x_2, \dots, x_n]$  such that there exists a polynomial in VP,  $g \in \mathbb{F}[x_1, \dots, x_n, y_1, \dots, y_m]$ , with  $m = \text{poly}(n)$  and,

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

It is easy to see  $VP \subseteq VNP$ .

# The Permanent

It is known that the determinant  $\in \text{VP}$ .

Permanent is similar to the determinant and is hard to compute (at least till now) and is thought to be in  $\text{VNP}$  (excluding  $\text{VP}$ ).



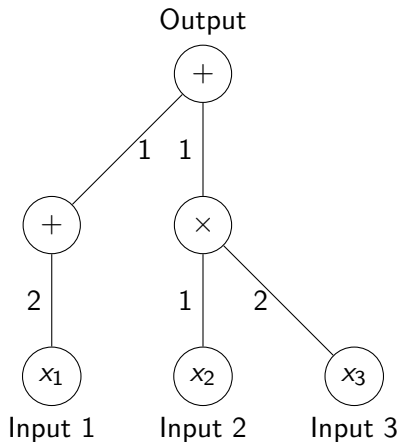
# Formula

## Formula

A formula is the same as a circuit with the constraint that any node present in the formula can have **at most one** outgoing edge.

These model the way we perform calculations on a piece of paper rather than a computer program.

## Example



In this example, the final output will be  $2x_1 + 2x_2x_3$

# Complexity Classes

## VF

The set of polynomials,  $f \in \mathbb{F}[x_1, x_2, \dots, x_n]$  that can be represented by a formula of size  $\text{poly}(n)$  is called **VF**.

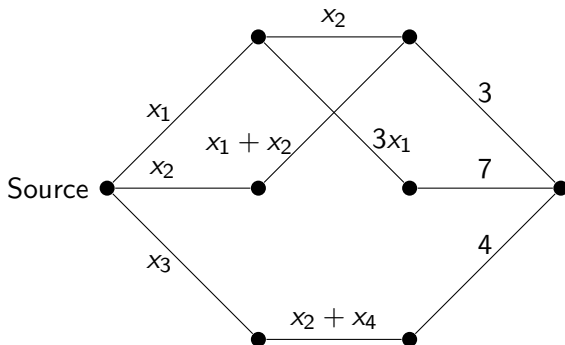
It is easy to see  $VF \subseteq VP$ .

# Algebraic Branching Programs

## ABP

An ABP is a *layered directed acyclic graph* with edges labelled with linear polynomials. There is a *source vertex* ( $s$ ) and a *sink vertex* ( $t$ ).

## Example



The final polynomial is calculated as

$$(x_3 \cdot (x_2 + x_4) \cdot 4) + (x_2 \cdot (x_1 + x_2) \cdot 3) + (x_1 \cdot x_2 \cdot 3) + (x_1 \cdot 3x_1 \cdot 7)$$

# Complexity Classes

## VBP

The class **VBP** is defined as the set of polynomials having size  $\text{poly}(n)$ .

It is known that  $VF \subseteq VBP \subseteq VP \subseteq VNP$  and these separations are hypothesised to be strict.

# Lower bounds on Sum of ABP's

Since finding general lower bounds has been proven to be hard, we will deal with finding lower bounds for *sum of special forms of ABP's* and relate them to general ABP's.

We will analyze **smABP's** and **ROABP's**

# Set-Multilinear ABP

Consider a partition of a set of variables  $\{X_1, X_2 \cdots X_d\}$ .

## Set-Multilinear

A *set-multilinear* polynomial is one in which is *homogeneous*, each variable has *individual degree at-most 1* (*multilinear*) and each monomial has *exactly one variable* from each of the  $d$  sets.



# read-once Oblivious ABP

An ABP is said to be *oblivious* if, for each layer, all the edge labels are *univariate* polynomials in a single variable.

An ABP is called a **read-once oblivious ABP (ROABP)** if each variable appears in at most one layer. [BDS23b]

# Hardness Bootstrapping

There is a theorem that shows how lower bounds on  $\sum smABP$  can lead to the separation of classes. It states,

## Theorem

**Theorem 1:** *Let  $n, d$  be integers such that  $d = O(\log n / \log \log n)$ . Let  $P_{n,d}$  be a set-multilinear polynomial in  $VNP$  of degree  $d$ . If  $P_{n,d}$  cannot be computed by a  $\sum smABP$  of width  $\text{poly}(n)$ , then  $VBP \neq VNP$ .*

[BDS23a]

# Outline

We will perform a sequence of structural transformations to the ABP.

- First *homogenize* the ABP, ie. we modify the ABP such that each vertex of the ABP computes a homogeneous polynomial.
- In addition, we will ensure that the ABP has  $d$  layers and all the edge labels are *linear forms*.
- Then we set-multilinearize the ABP. This step is efficient for low-degrees only as we obtain a sum of  $d^{O(d)}$  smABP's.

This means that superpolynomial bounds for  $\sum$ smABP imply the same for ABPs, albeit in the low degree regime.

# Simulating ABP using $\Sigma$ smABP

## Lemma 1

If a set-multilinear polynomial, with  $d$  sets each of size  $\leq n$  can be calculated by a ABP of size  $s$ , it can be calculated by a sum of smABP's of total width  $d^{O(d)}s$ . [BDS23b]

This can be directly used to prove **Theorem 1**.

# The proof

Firstly, using the assumption that it cannot be computed note that the width of the  $\Sigma$ smABP is,  $n^{\omega(1)}$ , and  $d^{O(d)}s \geq n^{\omega(1)}$ . Using the fact  $d = O(\log n / \log \log n)$  we get  $d^{O(d)} = \text{poly}(n)$ , hence  $s = n^{\omega(1)}$ , and we have our desired separation.

# Proof of Lemma 1

**Lemma 2:** *A degree  $d$  polynomial  $f$  that can be computed by an ABP of size  $s$  can also be computed by an homogeneous ABP of width  $s$  and length  $d$ . [BDS23b]*

**Lemma 3:** *If a homogenous ABP of width  $w$  and length  $d$  for a set-multilinear polynomial is needed, then the polynomial can be calculated by a  $\sum smABP$  of width  $d!w$ . [BDS23b]* These can be combined to prove the **Lemma 1**.

# Homogenization

To homogenize an ABP we will

- Divide each vertex,  $v$  into  $d$  vertices such that each vertex  $v^{(i)}$  computes the homogeneous part with degree  $i$ .
- Now, we will arrange all the vertex that compute the same degree in the same layer. This gives *width*  $s$  and length  $d$ , hence **Lemma 2**.
- This works because a vertex  $v^{(i)}$  can only have edges going to  $u^{(i)}$  or  $u^{(i+1)}$  (each edge is labelled by a linear form)

# ABP to $\sum$ smABP

We begin by writing the ABP for a sm-polynomial in its IMM form,

$$P_{n,d} = \prod_i^d M_i,$$

Here each  $M$  is a matrix with  $w \times w$  entries that are linear forms. Write each  $M_i = \sum_{j=1}^d M_{ij}$ , such that each  $M_{ij}$  has terms from the set  $X_j$ .

$$P_{n,d} = \prod_i^d \sum_{j=1}^d M_{ij}$$

[BDS23b]



# ABP to $\sum$ smABP

We can ignore the non-set-multilinear terms, i.e. ignore the products having terms like  $M_{ij}, M_{i'j}$ . This gives the form

$$P_{n,d} = \sum_{\pi \in S_d} \prod_{i=1}^d M_{i\pi(i)},$$

Hence we have a sum of  $d!$  smABP's each of width  $w$ .  
[BDS23b]

# Approximate Circuits...

There are a few polynomials that can be computed approximately in VP, if we just provide a little degree of freedom.

Instead of feeding polynomials in the inputs of the circuit we feed polynomials in  $\epsilon$  and then calculate the polynomial  $g \in \mathbb{F}[\epsilon][x_1, \dots, x_n]$ .

We say  $f \in \mathbb{F}[x_1, \dots, x_n]$  is approximated by  $g$  to an order of approximation  $M$  if

$$g(\mathbf{x}, \epsilon) = \epsilon^M f(\mathbf{x}) + \epsilon^{M+1} Q(\mathbf{x}, \epsilon)$$

# Approximate Circuits...

If the size of  $g$  over the field  $\mathbb{F}[\epsilon]$  is in  $\text{VP}$ , then  $f$  is in  $\overline{\text{VP}}$ .

If we restrict the class such that the polynomials in  $\epsilon$  used must have polynomial-sized circuits themselves, we get a new class called  $\overline{\text{VP}}_\epsilon$ . This is also called *presentable VP*.

Similar classes can be defined in the cases of  $\text{VNP}$ .

# Debordering VNP

It can be easily seen  $VP \subseteq \overline{VP}_\epsilon \subseteq \overline{VP}$ , and  $VNP \subseteq \overline{VNP}_\epsilon \subseteq \overline{VNP}$ . There is a theorem that relates these

**Theorem 2:** *Over any finite field  $VNP = \overline{VNP}_\epsilon$ .*

This gives new containments  $VP \subseteq \overline{VP}_\epsilon \subseteq VNP$ .

[BDS23a]

To prove this we will show  $VNP \subseteq \overline{VNP}_\epsilon$  and  $\overline{VNP}_\epsilon \subseteq VNP$ . The first containment is trivial; for showing the second one, we need some techniques like *Interpolation* and *Valiant's Criterion*.

# Interpolation

Consider a polynomial  $P(x_1, \dots, x_n, y)$  and  $\deg_y P = d$  then the polynomial can be written as

$$P(x_1, \dots, x_n, y) = P_0(x_1, \dots, x_n) + P_1(x_1, \dots, x_n)y + \dots + P_d(x_1, \dots, x_n)y^d$$

Consider  $d$  distinct constants  $\alpha_0, \dots, \alpha_d$  and let

$P(\alpha_i) = P(x_1, \dots, x_n, \alpha_i)$ . Each of the  $P_i$  can be calculated using the following matrix multiplication.

$$\begin{bmatrix} P_0 \\ P_1 \\ \vdots \\ P_d \end{bmatrix} = \begin{bmatrix} 1 & \alpha_0 & \dots & \alpha_0^d \\ 1 & \alpha_1 & \dots & \alpha_1^d \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha_d & \dots & \alpha_d^d \end{bmatrix}^{-1} \begin{bmatrix} P(\alpha_0) \\ P(\alpha_1) \\ \vdots \\ P(\alpha_d) \end{bmatrix}$$

[Sap15]

# #P/poly

In complexity theory,  $P/poly$  refers to the set of problems that can be solved using polynomial size circuits.

$\#P$  refers to the problem of finding the number of satisfying assignments for a problem.

Hence it is easy to understand that  $\#P/poly$  refers to the class of problems that are related to finding the number of satisfying assignments using small size circuits.

# Valian't Criterion

*Let  $f = \sum_e c_e \mathbf{x}^e$  be a polynomial in  $n$  variables of degree  $\text{poly}(n)$  over a field  $\mathbb{F}$ . Suppose that there exists a string function  $\phi : \{0, 1\}^* \rightarrow \{0, 1\}^*$  in  $\#P/\text{poly}$  such that  $\phi(\langle \mathbf{e} \rangle) = \langle c_e \rangle$ . Then, the polynomial  $f$  is in VNP over the field  $\mathbb{F}$ . [BDS23a]*

## Continuation of debordering...

Now that *interpolation* and *Valiant's Criterion* are covered, we can complete our proof. We have access to  $f \in \overline{\text{VNP}}_\epsilon$  using the approximation

$$g(\mathbf{x}, \epsilon) = \epsilon^M f(\mathbf{x}) + \epsilon^{M+1} Q(\mathbf{x}, \epsilon)$$

This is of the hypercube form,

$$g(\mathbf{x}, \epsilon) = \sum_{\mathbf{a} \in \{0,1\}^m} h(\mathbf{x}, \mathbf{a}, \epsilon)$$



- We will extract the coefficients of  $\epsilon^M x^e$  in  $g$  using *interpolation* and taking the interpolation points to be the *roots of unity*. Consequently  $c_e$  can be obtained as a hypercube sum of an *exponential degree* circuit of *polynomial size*
- Using finite field arithmetic and the closure of the Boolean class  $\#P$  under exponential sums, we can go to the *boolean world*
- Thus, we can show that the algebraic circuit above can be replaced by a (multi-output) *Boolean circuit* of polynomial size and the hypercube sum computing the coefficient function is demonstrated in  $\#P/poly$
- Now using Valiant's criterion we can claim  

$$\overline{VNP}_\epsilon \subseteq VNP \implies \overline{VNP}_\epsilon = VNP$$

[BDS23a]

# References



C.S. Bhargav, Prateek Dwivedi, and Nitin Saxena.

Learning the coefficients: A presentable version of border complexity and applications to circuit factoring.

2023.



C.S. Bhargav, Prateek Dwivedi, and Nitin Saxena.

Lower bounds for the sum of small-size algebraic branching programs.

2023.



Ramprasad Saptharishi.

A survey of lower bounds in arithmetic circuit complexity.

2015.