Structural Results for Arithmetic Formulas

Sébastien Tavenas
Univ. Savoie Mont Blanc, CNRS, LAMA
Joint work with

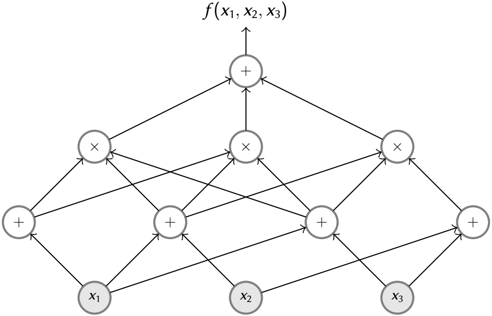
Hervé Fournier (Université Paris Cité)
Nutan Limaye (IT University of Copenhagen)
Guillaume Malod (Université Paris Cité)
Srikanth Srinivasan (University of Copenhagen)

June 03, 2025
CAALM Days 2025

Arithmetic Circuits

Let $P(x_1, \ldots, x_N) \in \mathbb{F}[x_1, \ldots, x_N]$ be a polynomial
(In this talk $\mathbb{F}$ field of characteristic 0)

An arithmetic circuit is a model of computation which computes polynomials.

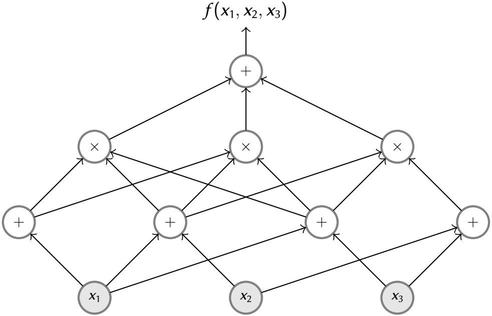


Hard to obtain lower bounds

Arithmetic Circuits

Let $P(x_1, \ldots, x_N) \in \mathbb{F}[x_1, \ldots, x_N]$ be a polynomial
(In this talk $\mathbb{F}$ field of characteristic 0)

An arithmetic circuit is a model of computation which computes polynomials.

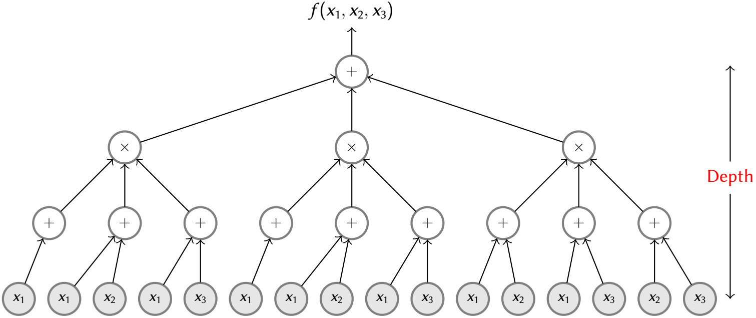


Hard to obtain lower bounds
First: lower bounds for formulas?

Arithmetic formula (graph is a tree)

Let $P(x_1, \ldots, x_N) \in \mathbb{F}[x_1, \ldots, x_N]$ be a polynomial

We can expand arithmetic circuits to get arithmetic formulas (possible blow-up: $(\text{size})^{\text{depth}}$).

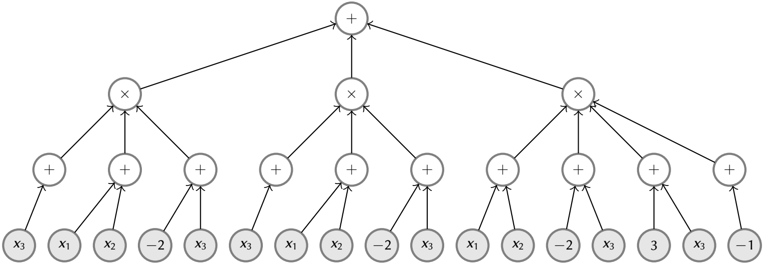


$x_1(x_1+x_2)(x_1+x_3)+x_1(x_1+x_2)(x_1+x_3)+(x_1+x_2)(x_1+x_3)(x_2+x_3)$

Size = Number of leaves. In this case 16

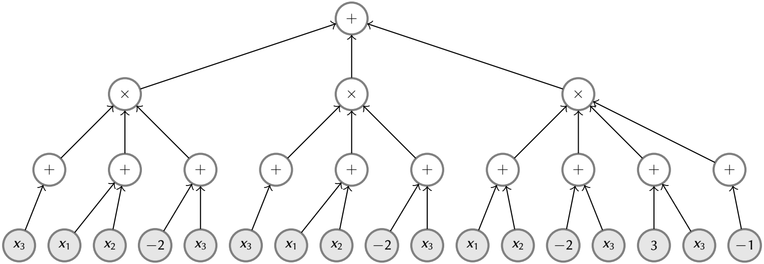
A formula is homogeneous if all intermediate gates are

Syntactic degree



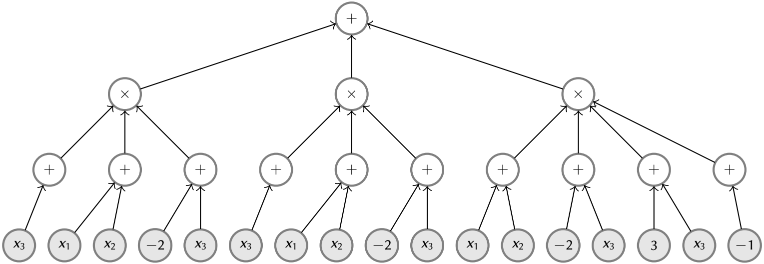
- Degree not clear from the formula (maybe some cancelations)
- Syntactic degree:
  1. $c \in \mathbb{F}$ has degree 0
  2. x_i has degree 1
  3. Degree of a $+$ gate: max of the degrees of the children
  4. Degree of a $*$ gate: sum of the degrees of the children
- If a formula is homogeneous:
  degree and syntactic degree coincide

Syntactic degree



- Degree not clear from the formula (maybe some cancelations)
- Syntactic degree:
  1. $c \in \mathbb{F}$ has degree 0
  2. x_i has degree 1
  3. Degree of a $+$ gate: max of the degrees of the children
  4. Degree of a $*$ gate: sum of the degrees of the children
- If a formula is homogeneous:
  degree and syntactic degree coincide

Syntactic degree



- Degree not clear from the formula (maybe some cancelations)
- Syntactic degree:
  1. $c \in \mathbb{F}$ has degree 0
  2. x_i has degree 1
  3. Degree of a $+$ gate: max of the degrees of the children
  4. Degree of a $*$ gate: sum of the degrees of the children
- If a formula is homogeneous:
  degree and syntactic degree coincide

Algebraic Complexity Theory

Algebraic lower bounds.

> **Question**
>
> Prove superpolynomial lower bounds for general arithmetic circuits.

Algebraic Complexity Theory

Algebraic lower bounds.

Question

Prove superpolynomial lower bounds for general arithmetic formulas.

Algebraic Complexity Theory

Algebraic lower bounds.

Question

Prove superpolynomial lower bounds for general arithmetic formulas.

- An $\Omega(n^2)$ lower bound for computing the elementary symmetric polynomials [Kal 1985, CKSV 2020]

Algebraic Complexity Theory

Algebraic lower bounds.

Question

Prove superpolynomial lower bounds for general arithmetic formulas.

- An $\Omega(n^2)$ lower bound for computing the elementary symmetric polynomials [Kal 1985, CKSV 2020]
- Superpolynomial lower bounds for constant depth arithmetic formulas for the determinant. [LST 2021]

Arithmetic formulas are surprisingly powerful

- Interpolation of coefficient of a polynomial of small degree d

$$[f(x)]_{x^p} = \sum_{i=0}^{d} \lambda_{pi} f(i)$$

- Elementary symmetric polynomials [Ben-Or]

$$e_d(x_1, \ldots, x_n) = \left[\prod_{i=1}^{n} (1 + tx_i)\right]_{t^d} \qquad \text{(not homogeneous!)}$$

- Polynomial division with remainder [BP 1985]
- Recently [AW 2024]: computing gcd of polynomials, discriminant, resultant, Bézout coefficients, square-free decomposition of a polynomial

Arithmetic formulas are surprisingly powerful

- Interpolation of coefficient of a polynomial of small degree d

$$[f(x)]_{x^p} = \sum_{i=0}^{d} \lambda_{pi} f(i)$$

- Elementary symmetric polynomials [Ben-Or]

$$e_d(x_1, \ldots, x_n) = \left[\prod_{i=1}^{n}(1 + tx_i)\right]_{t^d} \qquad \text{(not homogeneous!)}$$

- Polynomial division with remainder [BP 1985]
- Recently [AW 2024]: computing gcd of polynomials, discriminant, resultant, Bézout coefficients, square-free decomposition of a polynomial

- Product of n (2×2)-matrices can be done by poly-size formulas, but the depth has to be more than constant.

Arithmetic formulas are surprisingly powerful

- Interpolation of coefficient of a polynomial of small degree d

$$[f(x)]_{x^p} = \sum_{i=0}^{d} \lambda_{pi} f(i)$$

- Elementary symmetric polynomials [Ben-Or]

$$e_d(x_1, \ldots, x_n) = \left[\prod_{i=1}^{n}(1 + tx_i)\right]_{t^d} \qquad \text{(not homogeneous!)}$$

- Polynomial division with remainder [BP 1985]
- Recently [AW 2024]: computing gcd of polynomials, discriminant, resultant, Bézout coefficients, square-free decomposition of a polynomial

- Product of n (2×2)-matrices can be done by poly-size formulas, but the depth has to be more than constant.

- The determinant not expected to have poly-size formulas.

Algebraic Complexity Theory

Algebraic lower bounds.

Question

Prove superpolynomial lower bounds for general arithmetic formulas.

- An $\Omega(n^2)$ lower bound for computing the elementary symmetric polynomials [Kal 1985, CKSV 2020]
- Superpolynomial lower bounds for constant depth arithmetic formulas for the determinant. [LST 2021]

Algebraic Complexity Theory

Algebraic lower bounds.

Question

Prove superpolynomial lower bounds for general arithmetic formulas.

- An $\Omega(n^2)$ lower bound for computing the elementary symmetric polynomials [Kal 1985, CKSV 2020]
- Superpolynomial lower bounds for constant depth arithmetic formulas for the determinant. [LST 2021]
- More precisely: Superpolynomial lower bounds against arithmetic formulas of depth $O(\log \log d)$ for $d \ll s$.

What is the best general depth-reduction for formulas when $d \ll s$?

- Parallelization of the formulas to depth $O(\log s)$. [BKM73]
- Parallelization of the circuits to depth $O(\log d)$. [VSBR83]
  $\rightarrow$ Parallel. formulas to depth $O(\log d)$ and new size $d^{O(\log s)}$

What is the best general depth-reduction for formulas when $d \ll s$?

- Parallelization of the formulas to depth $O(\log s)$. [BKM73]
- Parallelization of the circuits to depth $O(\log d)$. [VSBR83]
  $\rightarrow$ Parallel. formulas to depth $O(\log d)$ and new size $d^{O(\log s)}$

Question:
Is there a poly size parallelization of formulas to depth $O(\log d)$?

What is the best general depth-reduction for formulas when $d \ll s$?

- Parallelization of the formulas to depth $O(\log s)$. [BKM73]
- Parallelization of the circuits to depth $O(\log d)$. [VSBR83]
  $\rightarrow$ Parallel. formulas to depth $O(\log d)$ and new size $d^{O(\log s)}$

Question:
Is there a poly size parallelization of formulas to depth $O(\log d)$?

- The question makes sense only for “unbounded fan-ins".
- If the sparsity of a polynomial is poly-size, then smal $\sum \prod$!
- If $d = o(\log s)$, we can homogeneize the formula [Raz13], which gives a better parallelization to depth $O(d) = o(\log s)$.

What is the best general depth-reduction for formulas when $d \ll s$?

- Parallelization of the formulas to depth $O(\log s)$. [BKM73]
- Parallelization of the circuits to depth $O(\log d)$. [VSBR83]
  $\rightarrow$ Parallel. formulas to depth $O(\log d)$ and new size $d^{O(\log s)}$

Question:
Is there a poly size parallelization of formulas to depth $O(\log d)$?

- The question makes sense only for “unbounded fan-ins".
- If the sparsity of a polynomial is poly-size, then smal $\sum\prod$!
- If $d = o(\log s)$, we can homogeneize the formula [Raz13], which gives a better parallelization to depth $O(d) = o(\log s)$.

What is the best general depth-reduction for formulas when $d \ll s$?

- Parallelization of the formulas to depth $O(\log s)$. [BKM73]
- Parallelization of the circuits to depth $O(\log d)$. [VSBR83]
  $\rightarrow$ Parallel. formulas to depth $O(\log d)$ and new size $d^{O(\log s)}$

Question:
Is there a poly size parallelization of formulas to depth $O(\log d)$?

- The question makes sense only for "unbounded fan-ins".
- If the sparsity of a polynomial is poly-size, then smal $\sum \prod$!
- If $d = o(\log s)$, we can homogeneize the formula [Raz13], which gives a better parallelization to depth $O(d) = o(\log s)$.

What is the best general depth-reduction for formulas when $d \ll s$?

- Parallelization of the formulas to depth $O(\log s)$. [BKM73]
- Parallelization of the circuits to depth $O(\log d)$. [VSBR83]
  $\rightarrow$ Parallel. formulas to depth $O(\log d)$ and new size $d^{O(\log s)}$

Question:
Is there a poly size parallelization of formulas to depth $O(\log d)$?

- [FLMST 2023] True for homogeneous formulas
- $\rightarrow$ weaker condition: syntactic degree polynomially bounded (Quasi-homogeneous)

Cost of (quasi-)homogenization

Circuits can be homogeneized

$\rightarrow$ size of the homogeneous formula $d^{O(\log s)}$

Case where poly-size homogeneization is known:

(1) $d = o(\log s)$ [Raz 2010], (2) Depth-3 [SW 2001, HY 2011]

Cost of (quasi-)homogenization

Circuits can be homogeneized

$\rightarrow$ size of the homogeneous formula $d^{O(\log s)}$

Case where poly-size homogeneization is known:

(1) $d = o(\log s)$ [Raz 2010], (2) Depth-3 [SW 2001, HY 2011]

[FLST 2024] P of degree d given by formula of size s:

- If $d = s^{o(1)}$, P has homogeneous formula of size $d^{o(\log s)}$
- $\forall \varepsilon > 0$, P has quasi-homogeneous formula of size $d^{o(\log s)}$ and syntactic degree $d^{1+\varepsilon}$

Cost of (quasi-)homogenization

Circuits can be homogeneized

$\to$ size of the homogeneous formula $d^{O(\log s)}$

Case where poly-size homogeneization is known:

(1) $d = o(\log s)$ [Raz 2010], (2) Depth-3 [SW 2001, HY 2011]

[FLST 2024] P of degree d given by formula of size s:

- If $d = s^{o(1)}$, P has homogeneous formula of size $d^{o(\log s)}$
- $\forall \varepsilon > 0$, P has quasi-homogeneous formula of size $d^{o(\log s)}$ and syntactic degree $d^{1+\varepsilon}$

Consequence: if P of degree n has no quasi-homogeneous formula of size $n^{o(\log n)}$, then P also does not have any formula of size $\mathrm{poly}(n)$

Combining Quasi-homogeneization + depth reduction

[VSBR 1983] Reduction to $O(\log d)$: cost $d^{O(\log s)}$

[FLMST 23]
Syntactic degree is $\text{poly}(d) \to$ parallelization depth $O(\log(d))$

Combining Quasi-homogeneization + depth reduction

[VSBR 1983] Reduction to $O(\log d)$: cost $d^{O(\log s)}$

[FLMST 23]
Syntactic degree is $\text{poly}(d) \to$ parallelization depth $O(\log(d))$

- If $d = o(\log s)$, (using [Raz13])
  the parallelization works for general formulas.

Combining Quasi-homogeneization + depth reduction

[VSBR 1983] Reduction to $O(\log d)$: cost $d^{O(\log s)}$

[FLMST 23]
Syntactic degree is $\text{poly}(d) \to$ parallelization depth $O(\log(d))$

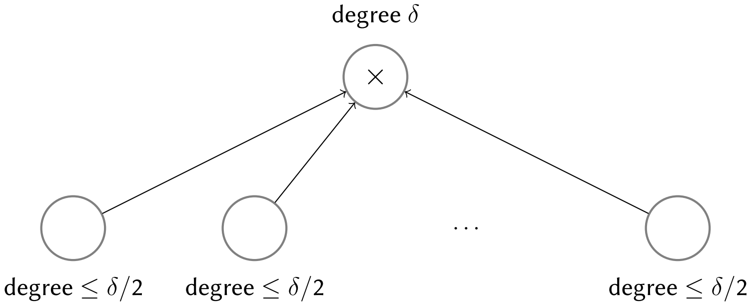
- If $d = o(\log s)$, (using [Raz13])
  the parallelization works for general formulas.

- [FLST 24] Quasi-homogeneous formula of size $d^{o(\log s)}$

  General parallel. to $O(\log d)$ but new size $d^{o(\log s)}$

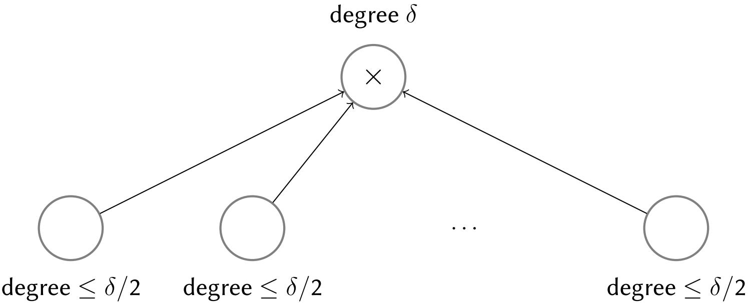
Parallelization of homogeneous formulas - Intuition I

If the (syntactic) degrees of all the gates are well-distributed:



Parallelization of homogeneous formulas - Intuition I

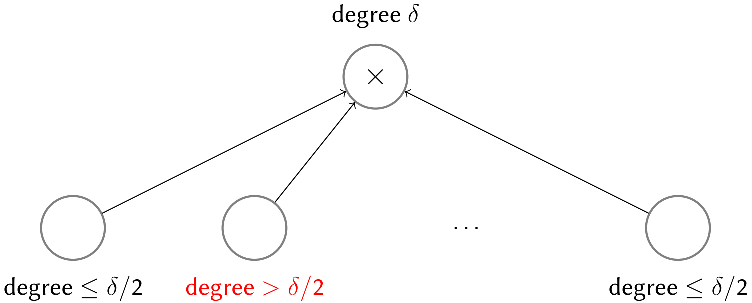
If the (syntactic) degrees of all the gates are well-distributed:



We directly obtain a formula of depth $O(\log d)$.

Parallelization of homogeneous formulas - Intuition I

If the (syntactic) degrees of all the gates are well-distributed:



We directly obtain a formula of depth $O(\log d)$.

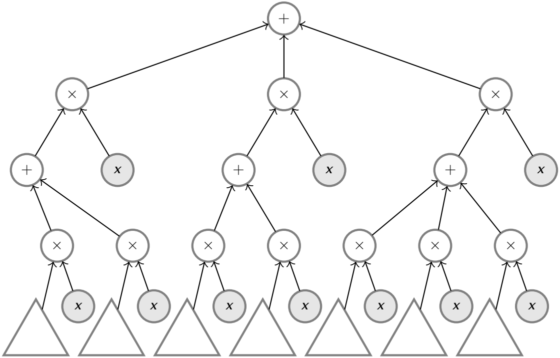
Hard case: Below a $\times$-gate, there is one subformula of large degree.

Parallelization of homogeneous formulas - Intuition II

OK, let us make a hard case!

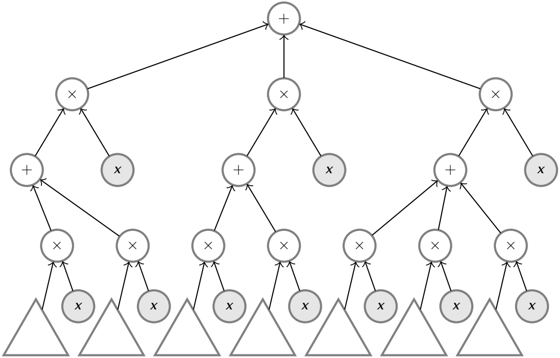
Parallelization of homogeneous formulas - Intuition II

OK, let us make a hard case!
Each multiplication is skew:



Parallelization of homogeneous formulas - Intuition II

OK, let us make a hard case!
Each multiplication is skew:



The sparsity is bounded by the number of leaves!
Small $\sum \prod$ formula!

Parallelization of homogeneous formulas - Proof

Let us associate to each node α its κ-potential:

$$\phi_\kappa(\alpha) = \lceil \log d_\alpha \rceil + \lceil \mathrm{depth}(F_\alpha)/\kappa \rceil.$$

Parallelization of homogeneous formulas - Proof

Let us associate to each node α its κ-potential:

$$\phi_\kappa(\alpha) = \lceil \log d_\alpha \rceil + \lceil \mathrm{depth}(F_\alpha)/\kappa \rceil.$$

Lemma

Let κ be a positive integer.
Any homogeneous formula F of fan-in 2, depth Δ, and size s can be parallelized into a formula G (of arbitrary fan-in) of product-depth at most $\phi_\kappa(root)$ and size at most $s2^{\kappa \log d}$.

Starting with [BKM73] parallelization.
Then, taking $\kappa = \lceil \log s / \log d \rceil$ gives the announced parallelization.

Proof of the lemma

$$\phi_\kappa(\alpha) = \lceil \log d_\alpha \rceil + \lceil \text{depth}(F_\alpha)/\kappa \rceil.$$

Lemma

Any homogeneous formula F of fan-in 2, sum-depth Δ, and size s can be parallelized into a formula G (of arbitrary fan-in) of product-depth at most $\phi_\kappa(\text{root})$ and size at most $s2^{\kappa \log d}$.

Proof of the lemma

$$\phi_\kappa(\alpha) = \lceil \log d_\alpha \rceil + \lceil \mathrm{depth}(F_\alpha)/\kappa \rceil.$$

Lemma

Any homogeneous formula F of fan-in 2, sum-depth Δ, and size s can be parallelized into a formula G (of arbitrary fan-in) of product-depth at most $\phi_\kappa(root)$ and size at most $s2^{\kappa \log d}$.

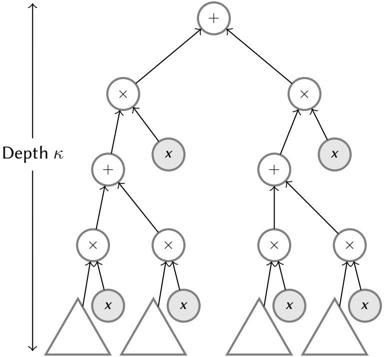
- Nodes α such that $\phi_\kappa(\alpha) = V$. G_V the corresponding formula.

Proof of the lemma

$$\phi_\kappa(\alpha) = \lceil \log d_\alpha \rceil + \lceil \text{depth}(F_\alpha)/\kappa \rceil.$$

Lemma

Any homogeneous formula F of fan-in 2, sum-depth Δ, and size s can be parallelized into a formula G (of arbitrary fan-in) of product-depth at most $\phi_\kappa(root)$ and size at most $s2^{\kappa \log d}$.



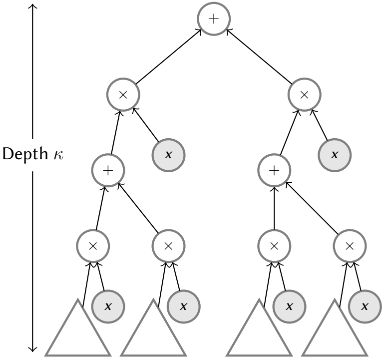
- Nodes α such that $\phi_\kappa(\alpha) = V$. G_V the corresponding formula.

Proof of the lemma

$$\phi_\kappa(\alpha) = \lceil \log d_\alpha \rceil + \lceil \mathrm{depth}(F_\alpha)/\kappa \rceil.$$

Lemma

Any homogeneous formula F of fan-in 2, sum-depth Δ, and size s can be parallelized into a formula G (of arbitrary fan-in) of product-depth at most $\phi_\kappa(root)$ and size at most $s2^{\kappa \log d}$.



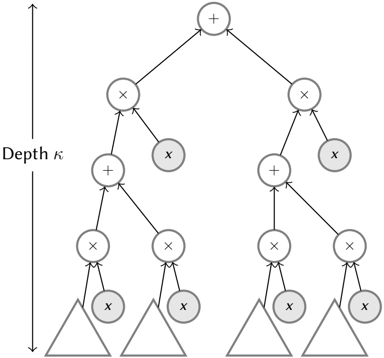
- Nodes α such that $\phi_\kappa(\alpha) = V$. G_V the corresponding formula.
- G_V is skew with fan-in 2, hence $G_V \in \sum^{2^\kappa} \prod$.

Proof of the lemma

$$\phi_\kappa(\alpha) = \lceil \log d_\alpha \rceil + \lceil \mathrm{depth}(F_\alpha)/\kappa \rceil.$$

Lemma

Any homogeneous formula F of fan-in 2, sum-depth Δ, and size s can be parallelized into a formula G (of arbitrary fan-in) of product-depth at most $\phi_\kappa(root)$ and size at most $s2^{\kappa \log d}$.



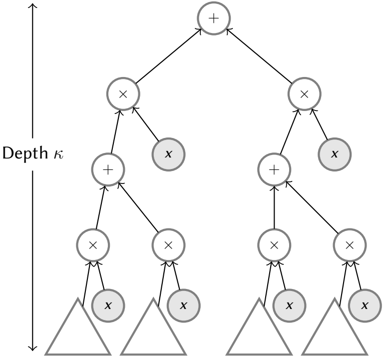
- Nodes α such that $\phi_\kappa(\alpha) = V$. G_V the corresponding formula.
- G_V is skew with fan-in 2, hence $G_V \in \sum^{2^\kappa} \prod$.
- Leaves of G_V: duplicated? Only if their log-degree decreases

Proof of the lemma

$$\phi_\kappa(\alpha) = \lceil \log d_\alpha \rceil + \lceil \text{depth}(F_\alpha)/\kappa \rceil.$$

Lemma

Any homogeneous formula F of fan-in 2, sum-depth Δ, and size s can be parallelized into a formula G (of arbitrary fan-in) of product-depth at most $\phi_\kappa(\text{root})$ and size at most $s2^{\kappa \log d}$.



- Nodes α such that $\phi_\kappa(\alpha) = V$. G_V the corresponding formula.
- G_V is skew with fan-in 2, hence $G_V \in \sum^{2^\kappa} \prod$.
- Leaves of G_V: duplicated? Only if their log-degree decreases
- Leaves of the whole formula duplicated at most $2^{\kappa \log d}$ times.

Summary

Any formula of size s and syntactic degree $\text{poly}(d)$
can be parallelized into a formula with
product-depth $\leq O(\log d)$ and size $\leq \text{poly}(s)$.

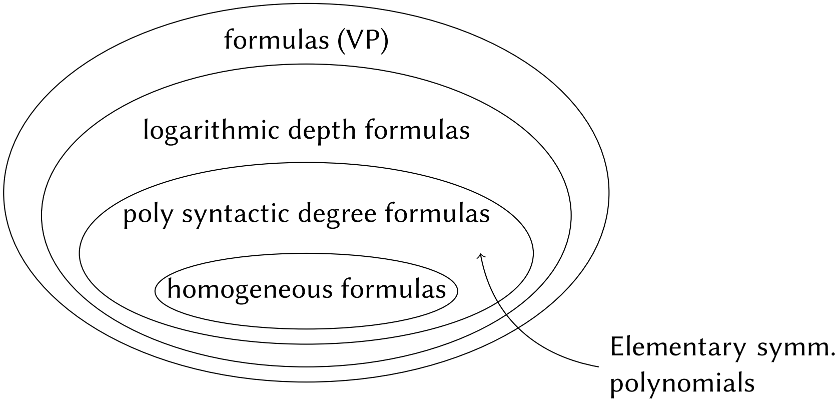
Corollary: Any formula of size s

(1) with $d = o(\log s)$, can be parallel.
to depth $\leq O(\log d)$ and size $\leq \text{poly}(s)$,

(2) can be parallel. to depth $\leq O(\log d)$ and size $\leq d^{o(\log s)}$

- It preserves monotonicity/order/(set)-multilinearity
- Optimal in the monotone case!
- Also obtain a near-linear parallelization [BB94,BCE95]:
size $s^{1+\varepsilon}$ and depth $2^{O(1/\varepsilon)} \log d$.

Open Questions



Thank you