A Quadratic Lower Bound against Algebraic Branching Programs

Prerona Chatterjee
Tata Instutute of Fundamental Research, Mumbai

with Mrinal Kumar (IITB), Adrian She (UoT), Ben Lee Volk (Caltech)

July 26, 2020

Algebraic Circuit Complexity

How succinctly can one represent a given polynomial?

Algebraic Circuit Complexity

How succinctly can one represent a given polynomial?

$$x^3 + 3x^2y + 3xy^2 + y^3$$

How succinctly can one represent a given polynomial?

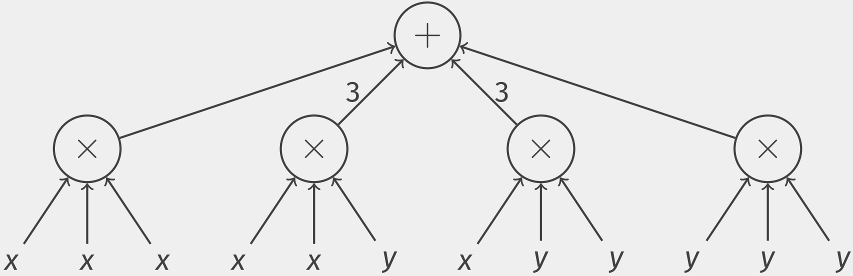
$$x^3 + 3x^2y + 3xy^2 + y^3$$

Sparse Representation

How succinctly can one represent a given polynomial?

$$x^3 + 3x^2y + 3xy^2 + y^3$$

Sparse Representation



$$x^3 + 3x^2y + 3xy^2 + y^3$$

$$x^3 + 3x^2y + 3xy^2 + y^3$$

Algebraic Formula (VF)

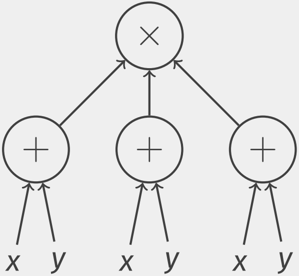
$(x + y) \cdot (x + y) \cdot (x + y)$

Algebraic Formulas and Algebraic Circuits

$$x^3 + 3x^2y + 3xy^2 + y^3$$

Algebraic Formula (VF)

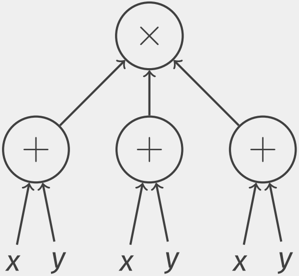
$$(x + y) \cdot (x + y) \cdot (x + y)$$



$$x^3 + 3x^2y + 3xy^2 + y^3$$

Algebraic Formula (VF)

$$(x + y) \cdot (x + y) \cdot (x + y)$$



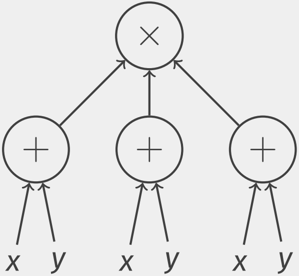
Algebraic Circuits (VP)

$$(x + y)^3$$

$$x^3 + 3x^2y + 3xy^2 + y^3$$

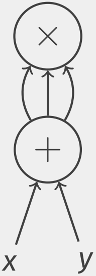
Algebraic Formula (VF)

$(x+y)\cdot(x+y)\cdot(x+y)$



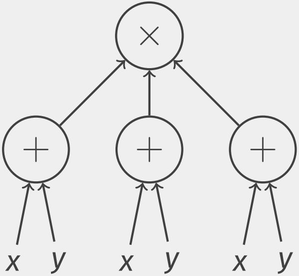
Algebraic Circuits (VP)

$(x+y)^3$



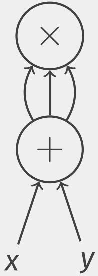
Algebraic Formula (VF)

$(x+y)\cdot(x+y)\cdot(x+y)$

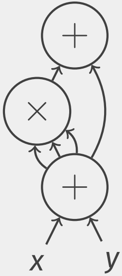


Algebraic Circuits (VP)

$(x+y)^3$



$(x+y)^3+(x+y)$

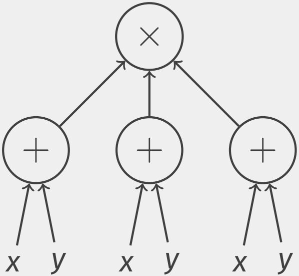


Algebraic Formulas and Algebraic Circuits

VF $\subseteq$ VP

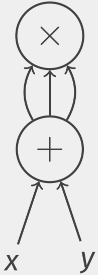
Algebraic Formula (VF)

$(x + y) \cdot (x + y) \cdot (x + y)$

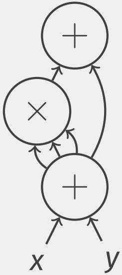


Algebraic Circuits (VP)

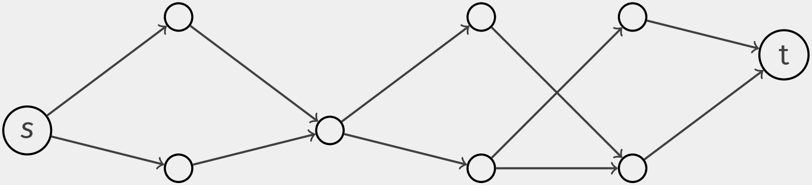
$(x + y)^3$

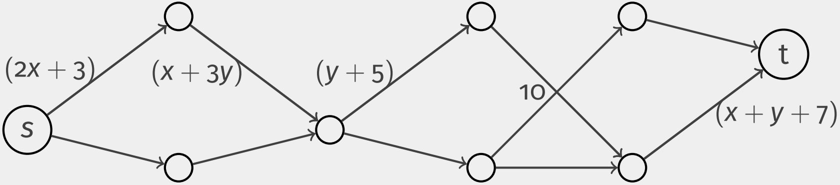


$(x + y)^3 + (x + y)$

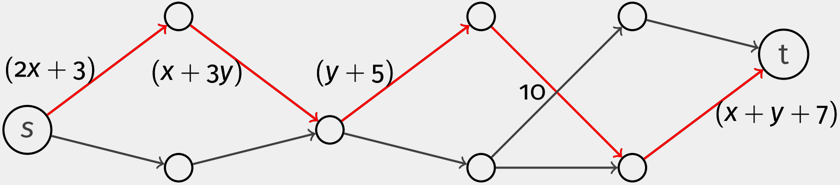


Algebraic Branching Programs

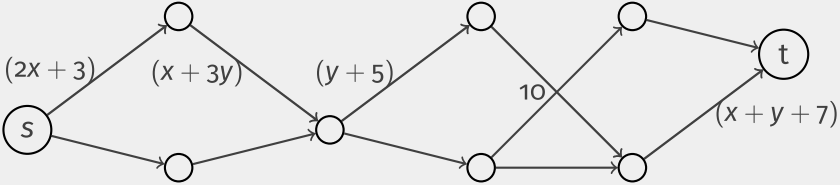




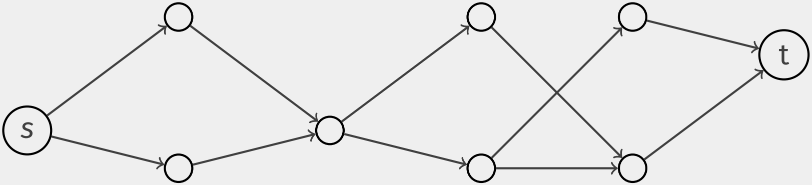
- Label on each edge: An affine linear form in $\{x_1, x_2, \ldots, x_n\}$



- Label on each edge: An affine linear form in $\{x_1, x_2, \ldots, x_n\}$
- Weight of path p = wt(p): Product of the edge labels on p



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



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

VF $\subseteq$ **VBP** $\subseteq$ **VP**

The Lower Bound Question

Q: Can one give a polynomial that is not succinctly representable?

The Lower Bound Question

Q: Can one give a polynomial that is not succinctly representable?

That is: Can one give an n-variate, degree d polynomial that can not be represented by a circuit/formula/ABP of size $\mathrm{poly}(n, d)$?

The Lower Bound Question

Q: Can one give a polynomial that is not succinctly representable?

That is: Can one give an n-variate, degree d polynomial that can not be represented by a circuit/formula/ABP of size $\mathrm{poly}(n, d)$?

Aim: Show a super-polynomial lower bound.

Q: Can one give a polynomial that is not succinctly representable?

That is: Can one give an n-variate, degree d polynomial that can not be represented by a circuit/formula/ABP of size $\mathrm{poly}(n, d)$?

Aim: Show a super-polynomial lower bound.

For Circuits [Baur-Strassen]:
Any fan-in 2 circuit computing $\sum_{i=1}^{n} x_i^d$ requires $\Omega(n \log d)$ multiplication gates.

The Lower Bound Question

Q: Can one give a polynomial that is not succinctly representable?

That is: Can one give an n-variate, degree d polynomial that can not be represented by a circuit/formula/ABP of size $\mathrm{poly}(n, d)$?

Aim: Show a super-polynomial lower bound.

For Circuits [Baur-Strassen]:
Any fan-in 2 circuit computing $\sum_{i=1}^{n} x_i^d$ requires $\Omega(n \log d)$ multiplication gates.

For Formulas [Kalorkoti]:
Any formula computing $\mathrm{Det}_{n \times n}$ requires $\Omega(n^3)$ wires.

General ABPs [Baur-Strassen]:
Any ABP computing $\sum_{i=1}^{n} x_i^d$ requires $\Omega(n \log d)$ wires.

General ABPs [Baur-Strassen]:
Any ABP computing $\sum_{i=1}^{n} x_i^d$ requires $\Omega(n \log d)$ wires.

Restricted ABPs [Kumar]:
Any ABP with $(d+1)$ layers computing $\sum_{i=1}^{n} x_i^d$ has $\Omega(nd)$ vertices.

Our Main Result

General ABPs [Baur-Strassen]:
Any ABP computing $\sum_{i=1}^{n} x_i^d$ requires $\Omega(n \log d)$ wires.

Restricted ABPs [Kumar]:
Any ABP with $(d+1)$ layers computing $\sum_{i=1}^{n} x_i^d$ has $\Omega(nd)$ vertices.

Our Main Result:
Any ABP computing $\sum_{i=1}^{n} x_i^d$ requires $\Omega(nd)$ vertices.

Step 0 ([Kumar]): Look at the homogeneous case

Any ABP with $(d+1)$ layers computing $\sum_{i=1}^{n} x_i^d$ has $\Omega(nd)$ vertices.

Step 0 ([Kumar]): Look at the homogeneous case

Any ABP with $(d+1)$ layers computing $\sum_{i=1}^{n} x_i^d$ has $\Omega(nd)$ vertices.

Step 1: Generalise above statement to get the base case

Any ABP with $(d+1)$ layers

How do we show this?

Step 0 ([Kumar]): Look at the homogeneous case

Any ABP with $(d+1)$ layers computing $\sum_{i=1}^{n} x_i^d$ has $\Omega(nd)$ vertices.

Step 1: Generalise above statement to get the base case

Any ABP with $(d+1)$ layers computing a polynomial of the form

$$f = \sum_{i=1}^{n} x_i^d + \sum_{i=1}^{r} A_i(\mathbf{x}) \cdot B_i(\mathbf{x}) + \delta(\mathbf{x})$$

Step 0 ([Kumar]): Look at the homogeneous case

Any ABP with $(d+1)$ layers computing $\sum_{i=1}^{n} x_i^d$ has $\Omega(nd)$ vertices.

Step 1: Generalise above statement to get the base case

Any ABP with $(d+1)$ layers computing a polynomial of the form

$$f = \sum_{i=1}^{n} x_i^d + \sum_{i=1}^{r} A_i(\mathbf{x}) \cdot B_i(\mathbf{x}) + \delta(\mathbf{x})$$

where $\quad A_i(0) = 0 = B_i(0) \quad$ and $\quad \deg(\delta(\mathbf{x})) < d$,

Step 0 ([Kumar]): Look at the homogeneous case

Any ABP with $(d+1)$ layers computing $\sum_{i=1}^{n} x_i^d$ has $\Omega(nd)$ vertices.

Step 1: Generalise above statement to get the base case

Any ABP with $(d+1)$ layers computing a polynomial of the form

$$f = \sum_{i=1}^{n} x_i^d + \sum_{i=1}^{r} A_i(\mathbf{x}) \cdot B_i(\mathbf{x}) + \delta(\mathbf{x})$$

where $A_i(0) = 0 = B_i(0)$ and $\deg(\delta(\mathbf{x})) < d$, has at least

$$((n/2) - r) \cdot (d-1) \quad \text{vertices.}$$

How do we show this?

Step 2: Iteratively reduce to Base Case

In each iteration, reduce the number of layers till it becomes $(d+1)$ such that

Step 2: Iteratively reduce to Base Case

In each iteration, reduce the number of layers till it becomes $(d+1)$ such that

- the number of layers is reduced by a constant fraction,

Step 2: Iteratively reduce to Base Case

In each iteration, reduce the number of layers till it becomes $(d+1)$ such that

- the number of layers is reduced by a constant fraction,
- the size does not increase,

How do we show this?

Step 2: Iteratively reduce to Base Case

In each iteration, reduce the number of layers till it becomes $(d+1)$ such that

- the number of layers is reduced by a constant fraction,
- the size does not increase,
- the polynomial being computed continues to look like

$$f_{\ell+1} = \sum_{i=1}^{n} x_i^d + \sum_{i=1}^{r_{\ell+1}} A_i(\mathbf{x}) \cdot B_i(\mathbf{x}) + \delta_{\ell+1}(\mathbf{x})$$

where $\quad A_i(0) = 0 = B_i(0) \quad$ and $\quad \deg(\delta_{\ell+1}(\mathbf{x})) < d$,

Step 2: Iteratively reduce to Base Case

In each iteration, reduce the number of layers till it becomes $(d+1)$ such that

- the number of layers is reduced by a constant fraction,
- the size does not increase,
- the polynomial being computed continues to look like

$$f_{\ell+1} = \sum_{i=1}^{n} x_i^d + \sum_{i=1}^{r_{\ell+1}} A_i(\mathbf{x}) \cdot B_i(\mathbf{x}) + \delta_{\ell+1}(\mathbf{x})$$

  where $A_i(0) = 0 = B_i(0)$ and $\deg(\delta_{\ell+1}(\mathbf{x})) < d$,
- number of error terms collected is small.

The Induction Step

ℓ-th step

The Induction Step

ℓ-th step

Given: ABP $\mathcal{A}_\ell$ of size = s_ℓ
no. of layers = d_ℓ
no. of error terms = r_ℓ

ℓ-th step

Given: ABP $\mathcal{A}_\ell$ of size = s_ℓ
no. of layers = d_ℓ
no. of error terms = r_ℓ

Want to construct: ABP $\mathcal{A}_{\ell+1}$ of size = $s_{\ell+1}$

ℓ-th step

Given: ABP $\mathcal{A}_\ell$ of size = s_ℓ
no. of layers = d_ℓ
no. of error terms = r_ℓ

Want to construct: ABP $\mathcal{A}_{\ell+1}$ of size = $s_{\ell+1} \leq s_\ell$

ℓ-th step

Given: ABP $\mathcal{A}_\ell$ of size = s_ℓ
no. of layers = d_ℓ
no. of error terms = r_ℓ

Want to construct: ABP $\mathcal{A}_{\ell+1}$ of size = $s_{\ell+1} \leq s_\ell$
no. of layers = $d_{\ell+1}$

ℓ-th step

Given: ABP $\mathcal{A}_\ell$ of size = s_ℓ
no. of layers = d_ℓ
no. of error terms = r_ℓ

Want to construct: ABP $\mathcal{A}_{\ell+1}$ of size = $s_{\ell+1} \leq s_\ell$
no. of layers = $d_{\ell+1} \leq \frac{2}{3} d_\ell$

ℓ-th step

Given: ABP $\mathcal{A}_\ell$ of size = s_ℓ
no. of layers = d_ℓ
no. of error terms = r_ℓ

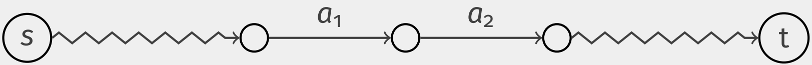
Want to construct: ABP $\mathcal{A}_{\ell+1}$ of size = $s_{\ell+1} \leq s_\ell$
no. of layers = $d_{\ell+1} \leq \frac{2}{3} d_\ell$
no. of error terms = $r_{\ell+1}$

ℓ-th step

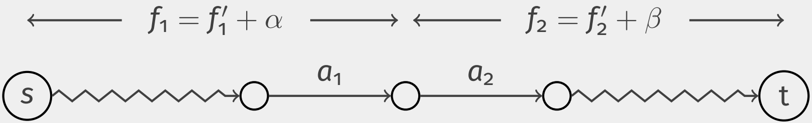
Given: ABP $\mathcal{A}_\ell$ of size = s_ℓ
no. of layers = d_ℓ
no. of error terms = r_ℓ

Want to construct: ABP $\mathcal{A}_{\ell+1}$ of size = $s_{\ell+1} \leq s_\ell$
no. of layers = $d_{\ell+1} \leq \frac{2}{3} d_\ell$
no. of error terms = $r_{\ell+1} \leq r_\ell + \frac{s_\ell}{d_\ell/3}$

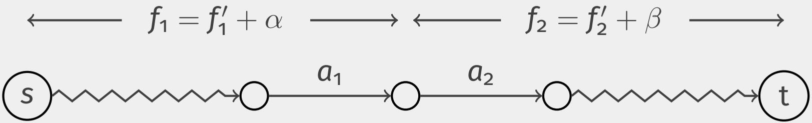
Proof of the Induction Step



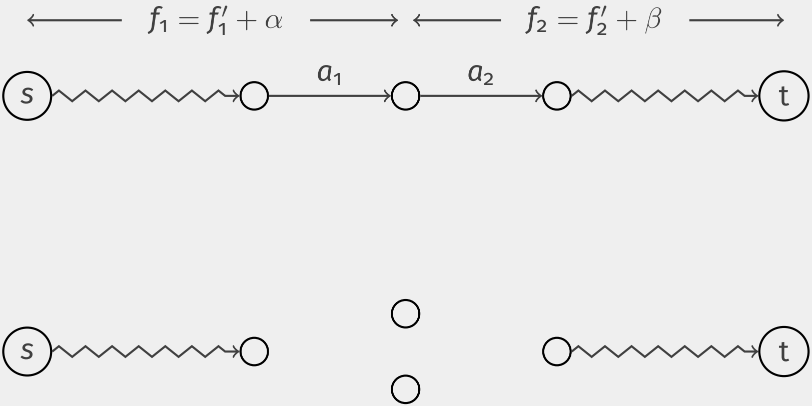
Proof of the Induction Step



$\mathcal{A}_\ell = f_1 \cdot f_2$

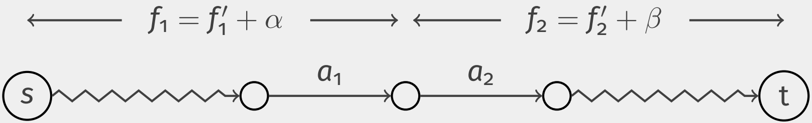


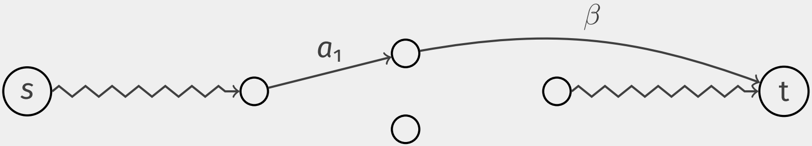
$\mathcal{A}_\ell = f_1 \cdot f_2$



Proof of the Induction Step

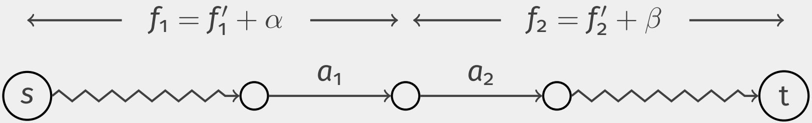
$\mathcal{A}_\ell = f_1 \cdot f_2$

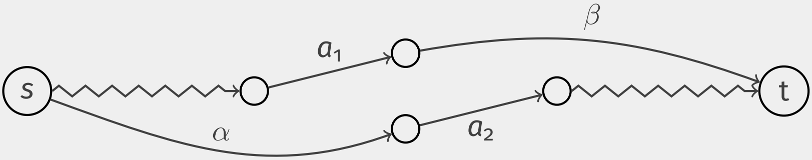




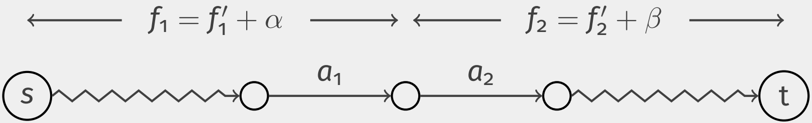
Proof of the Induction Step

$\mathcal{A}_\ell = f_1 \cdot f_2$

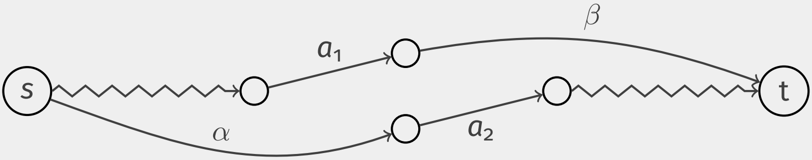




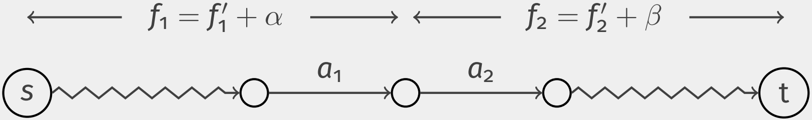
$\mathcal{A}_\ell = f_1 \cdot f_2$



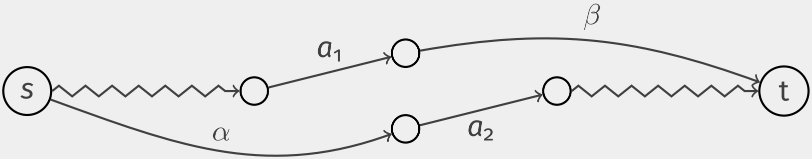
$\mathcal{A}_{\ell+1} = \beta \cdot f_1 + \alpha \cdot f_2$



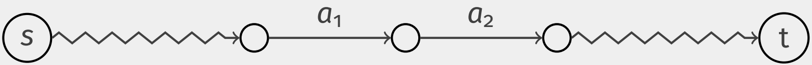
$\mathcal{A}_\ell = f_1 \cdot f_2$

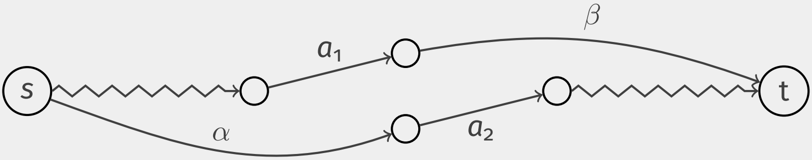


$\mathcal{A}_{\ell+1} = \beta \cdot f_1 + \alpha \cdot f_2 = \mathcal{A}_\ell - f_1' \cdot f_2' + \alpha \cdot \beta$

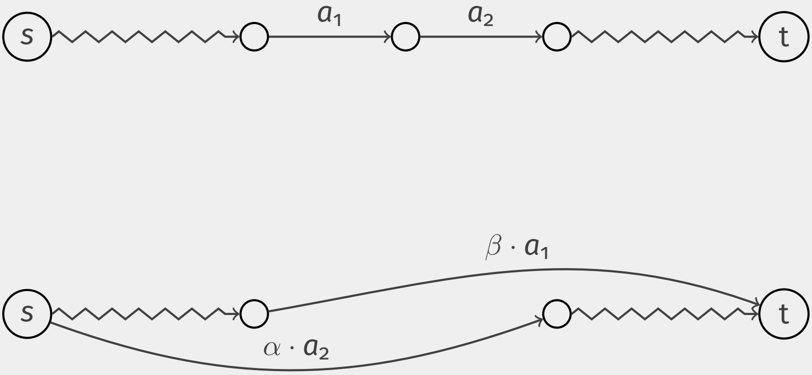


Proof of the Induction Step

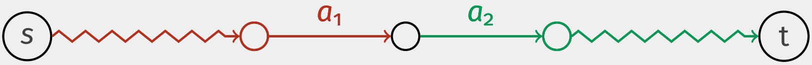


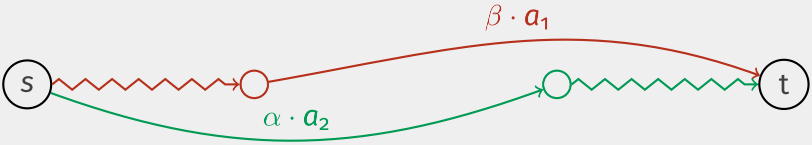


Proof of the Induction Step



Proof of the Induction Step





Other Results for ABPs

1. If the edge labels on the ABP are allowed to have degree Δ, then the lower bound we get is $\Omega(n^2/\Delta)$.

Other Results for ABPs

1. If the edge labels on the ABP are allowed to have degree Δ, then the lower bound we get is $\Omega(n^2/\Delta)$.

2. For unlayered ABPs with edge labels of degree $\leq \Delta$, the lower bound we get is $\Omega(n \log n/\Delta \log\log n)$.

1. If the edge labels on the ABP are allowed to have degree Δ, then the lower bound we get is $\Omega(n^2/\Delta)$.
2. For unlayered ABPs with edge labels of degree $\leq \Delta$, the lower bound we get is $\Omega(n \log n / \Delta \log \log n)$.
3. The lower bound is also true for a multilinear polynomial

$$\mathrm{ESym}(n, 0.1n) = \sum_{i_1 < \cdots < i_{0.1n} \in [n]} \prod_{j=1}^{0.1n} x_{i_j}.$$

[Kalorkoti]: Any formula computing $\mathrm{Det}_{n \times n}$ requires $\Omega(n^3)$ wires.

A lower bound for formulas

[Kalorkoti]: Any formula computing $\mathrm{Det}_{n\times n}$ requires $\Omega(n^3)$ wires.

[Shpilka, Yehudayoff] (using Kalorkoti's method):
Any formula computing $\sum_{i=1}^{n}\sum_{j=1}^{n} x_i^j y_j$ requires $\Omega(n^2)$ wires.

[Kalorkoti]: Any formula computing $\mathrm{Det}_{n\times n}$ requires $\Omega(n^3)$ wires.

[Shpilka, Yehudayoff] (using Kalorkoti's method):
Any formula computing $\sum_{i=1}^{n}\sum_{j=1}^{n} x_i^j y_j$ requires $\Omega(n^2)$ wires.

Interesting Case: Multilinear polynomials

- Trival to get $\Omega(nd)$ for n-variate, i-degree d polynomials.

A lower bound for formulas

[Kalorkoti]: Any formula computing $\mathrm{Det}_{n\times n}$ requires $\Omega(n^3)$ wires.

[Shpilka, Yehudayoff] (using Kalorkoti's method):
Any formula computing $\sum_{i=1}^{n}\sum_{j=1}^{n} x_i^j y_j$ requires $\Omega(n^2)$ wires.

Interesting Case: Multilinear polynomials

- Trival to get $\Omega(nd)$ for n-variate, i-degree d polynomials.
- Multilinearising the SY polynomial gives $\Omega(n^2/\log n)$.

A lower bound for formulas

[Kalorkoti]: Any formula computing $\mathrm{Det}_{n\times n}$ requires $\Omega(n^3)$ wires.

[Shpilka, Yehudayoff] (using Kalorkoti's method):
Any formula computing $\sum_{i=1}^{n}\sum_{j=1}^{n} x_i^j y_j$ requires $\Omega(n^2)$ wires.

Interesting Case: Multilinear polynomials

- Trival to get $\Omega(nd)$ for n-variate, i-degree d polynomials.
- Multilinearising the SY polynomial gives $\Omega(n^2/\log n)$.
- Kalorkoti's method can not give better than $\Omega(n^2/\log n)$.

A lower bound for formulas

[Kalorkoti]: Any formula computing $\mathrm{Det}_{n\times n}$ requires $\Omega(n^3)$ wires.

[Shpilka, Yehudayoff] (using Kalorkoti's method):
Any formula computing $\sum_{i=1}^{n}\sum_{j=1}^{n} x_i^j y_j$ requires $\Omega(n^2)$ wires.

Interesting Case: Multilinear polynomials

- Trival to get $\Omega(nd)$ for n-variate, i-degree d polynomials.
- Multilinearising the SY polynomial gives $\Omega(n^2/\log n)$.
- Kalorkoti's method can not give better than $\Omega(n^2/\log n)$.

Our Result: Any formula computing $\mathrm{ESym}_{n,0.1n}$ has $\Omega(n^2)$ vertices, where

$$\mathrm{ESym}(n, 0.1n) = \sum_{i_1<\cdots<i_{0.1n}\in[n]} \prod_{j=1}^{0.1n} x_{i_j}.$$

1. Prove a quadratic lower bound (on wires?) for un-layered Algebraic Branching Programs

Open Threads

1. Prove a quadratic lower bound (on wires?) for un-layered Algebraic Branching Programs
2. Prove a super-quadratic lower bound on (homogeneous?) formulas (of constant depth?)

1. Prove a quadratic lower bound (on wires?) for un-layered Algebraic Branching Programs
2. Prove a super-quadratic lower bound on (homogeneous?) formulas (of constant depth?)
3. Reprove the $\Omega(n \log n)$ lower bound for general circuits

Open Threads

1. Prove a quadratic lower bound (on wires?) for un-layered Algebraic Branching Programs
2. Prove a super-quadratic lower bound on (homogeneous?) formulas (of constant depth?)
3. Reprove the $\Omega(n \log n)$ lower bound for general circuits

Thank you!