ALGEBRAIC COMPLEXITY: AN INTRODUCTION

SRIKANTH SRINIVASAN
(AARHUS UNIVERSITY)

Polynomials

$$P_n(x_1, \ldots, x_n) \in F[x_1, \ldots, x_n] \quad , \quad n \geq 1$$

$$P_n(\bar{x}) = \sum_{\bar{e} = (e_1, \ldots, e_n)} \alpha_{\bar{e}} \, x^{\bar{e}}$$

Polynomials

$$P_n(x_1, \ldots, x_n) \in \mathbb{F}[x_1, \ldots, x_n], \quad n \geq 1$$

$$P_n(\bar{x}) = \sum_{\bar{e} = (e_1, \ldots, e_n)} \alpha_{\bar{e}} \, x^{\bar{e}}$$

Defines a computational problem

Input: $a \in \mathbb{F}^n$

Output: $P_n(a) \in \mathbb{F}$

Polynomials

$$P_n(x_1, \ldots, x_n) \in \mathbb{F}[x_1, \ldots, x_n], \quad n \geq 1$$

$$P_n(\bar{x}) = \sum_{\bar{e} = (e_1, \ldots, e_n)} \alpha_{\bar{e}} \, x^{\bar{e}}$$

Defines a computational problem

Input: $a \in \mathbb{F}^n$

Output: $P_n(a) \in \mathbb{F}$

Captures many interesting computational problems.

Examples

→ $\det_m(x_{i,j} : i,j \in [m]) = \sum_{\sigma \in S_m} \operatorname{sgn}(\sigma) \prod_{i=1}^{m} x_{i,\sigma(i)}$

Examples

→ $\det_m(x_{i,j} : i,j \in [m]) = \sum_{\sigma \in S_m} \operatorname{sgn}(\sigma) \prod_{i=1}^{m} x_{i,\sigma(i)}$

→ $\operatorname{per}_m(x_{i,j} : i,j \in [m]) = \sum_{\sigma \in S_m} \prod_{i=1}^{m} x_{i,\sigma(i)}$

Examples

→ $\det_m(x_{i,j} : i,j \in [m]) = \sum_{\sigma \in S_m} \operatorname{sgn}(\sigma) \prod_{i=1}^{m} x_{i,\sigma(i)}$

→ $\operatorname{per}_m(x_{i,j} : i,j \in [m]) = \sum_{\sigma \in S_m} \prod_{i=1}^{m} x_{i,\sigma(i)}$

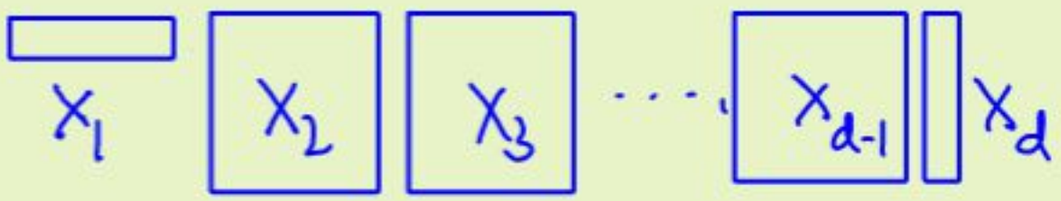
→ $E_n^d(x_1, \ldots, x_n) = \sum_{S \subseteq [n],\, |S|=d} \prod_{i \in S} x_i$

Examples

→ $\det_m(x_{i,j} : i,j \in [m]) = \sum_{\sigma \in S_m} sgn(\sigma) \prod_{i=1}^{m} x_{i,\sigma(i)}$

→ $per_m(x_{i,j} : i,j \in [m]) = \sum_{\sigma \in S_m} \prod_{i=1}^{m} x_{i,\sigma(i)}$

→ $E_n^d(x_1, \ldots, x_n) = \sum_{S \subseteq [n],\, |S|=d} \prod_{i \in S} x_i$

→ $IMM_n^d(X_1, \ldots, X_d) =$ 

Two Caveats

Two Caveats

→ Might refer to $P(x_1, \ldots, x_n)$ without reference to $(P_n)_{n \geq 1}$.

Two Caveats

→ Might refer to $P(x_1, \ldots, x_n)$ without reference to $(P_n)_{n \geq 1}$.

→ Typically assume that $\deg(P_n) \leq n^{O(1)}$.

Two Caveats

→ Might refer to $P(x_1, \ldots, x_n)$ without reference to $(P_n)_{n \geq 1}$.

→ Typically assume that $\deg(P_n) \leq n^{O(1)}$.

→ Theory becomes cleaner.

Two Caveats

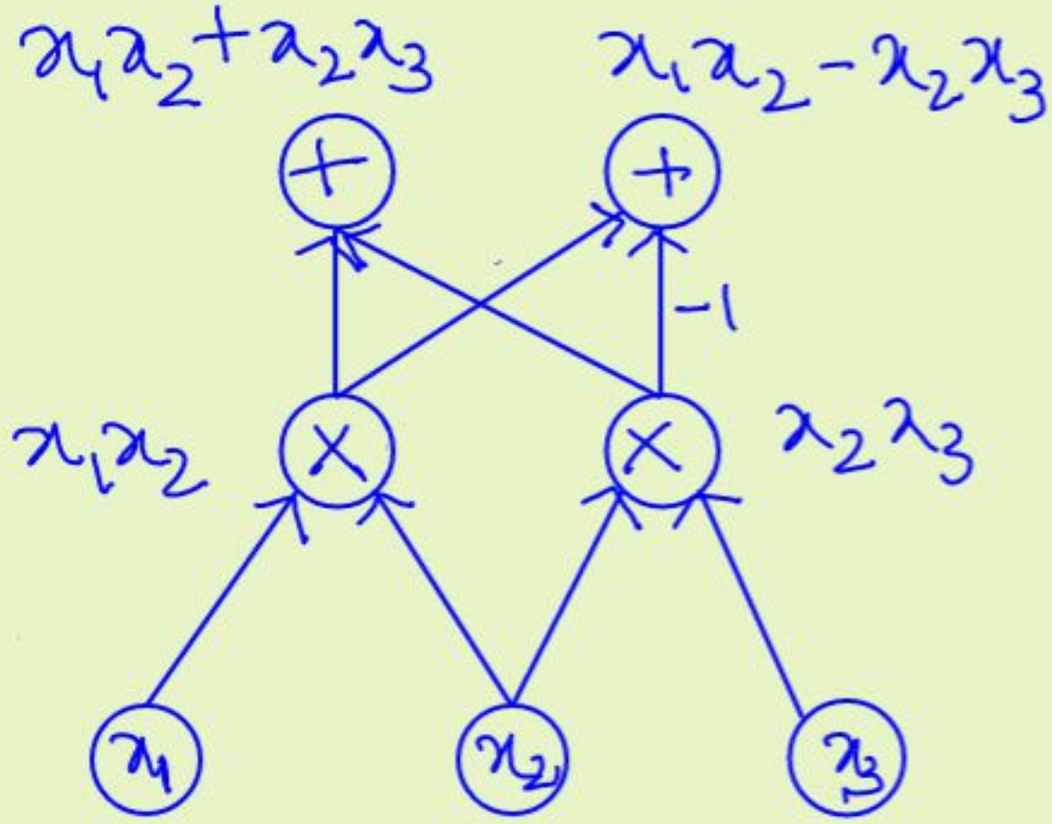
→ Might refer to $P(x_1, \ldots, x_n)$ without reference to $(P_n)_{n \geq 1}$.

→ Typically assume that $\deg(P_n) \leq n^{O(1)}$.

→ Theory becomes cleaner.

→ True for many (most?) naturally occurring problems.

Algebraic circuits



→ One or more output gates.

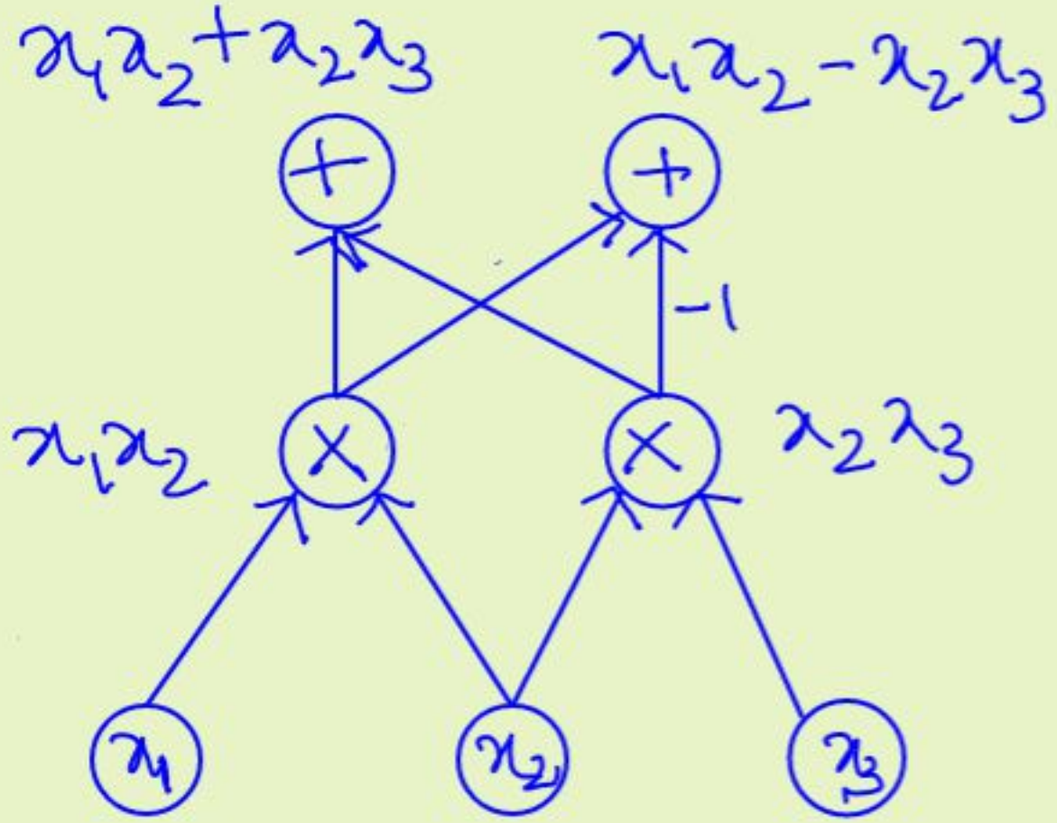
→ labelled DAG

→ Leaves ↔ variables, constants

→ × - gates for product

→ + - gates for sums (linear combinations)

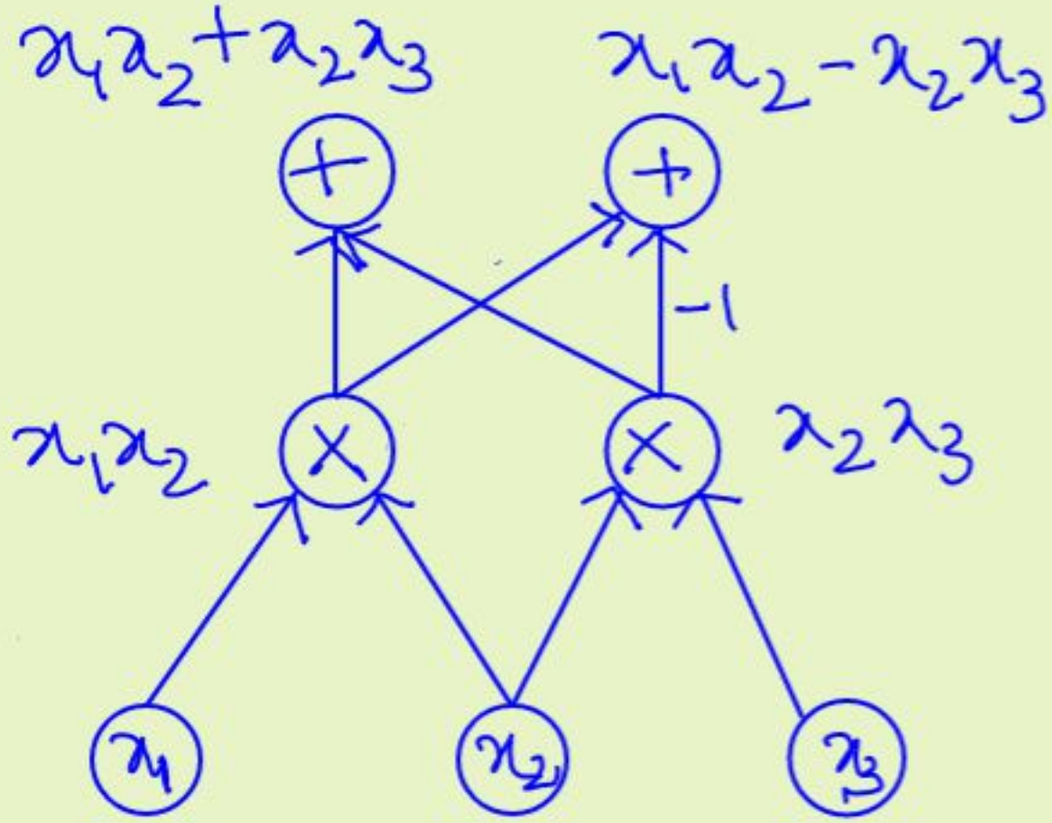
Algebraic circuits



Size = # of edges/wires

Depth = length of longest input-output path.

Algebraic circuits



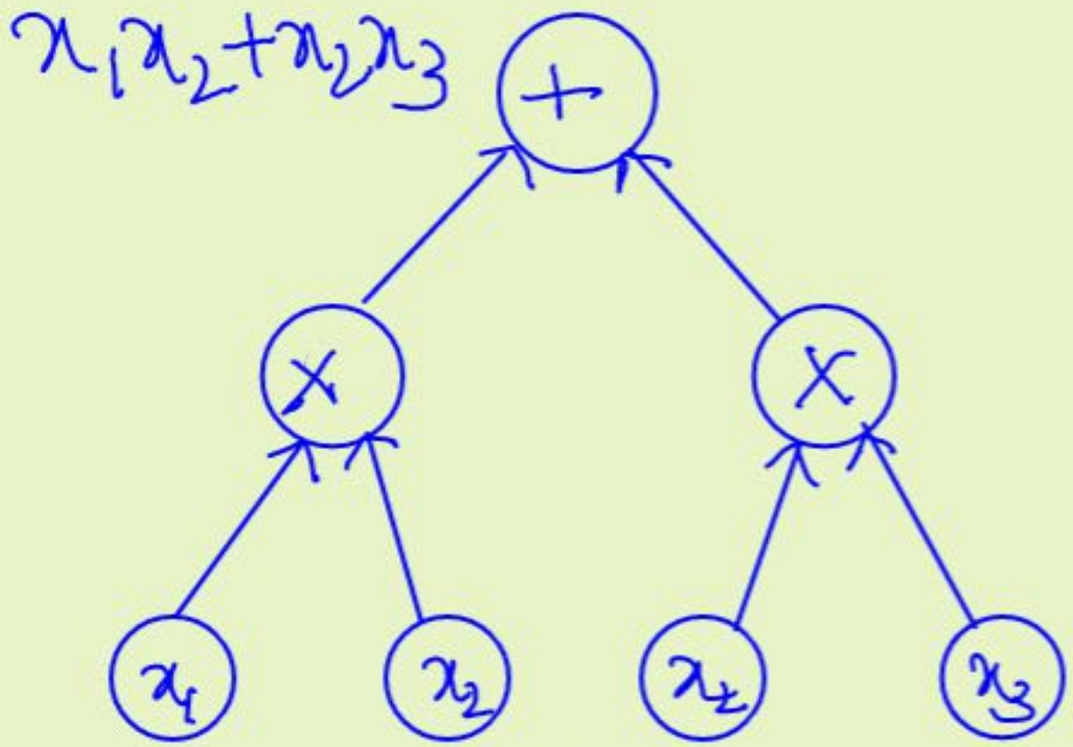
Size = 8

Depth = 2

Size = # of edges/wires

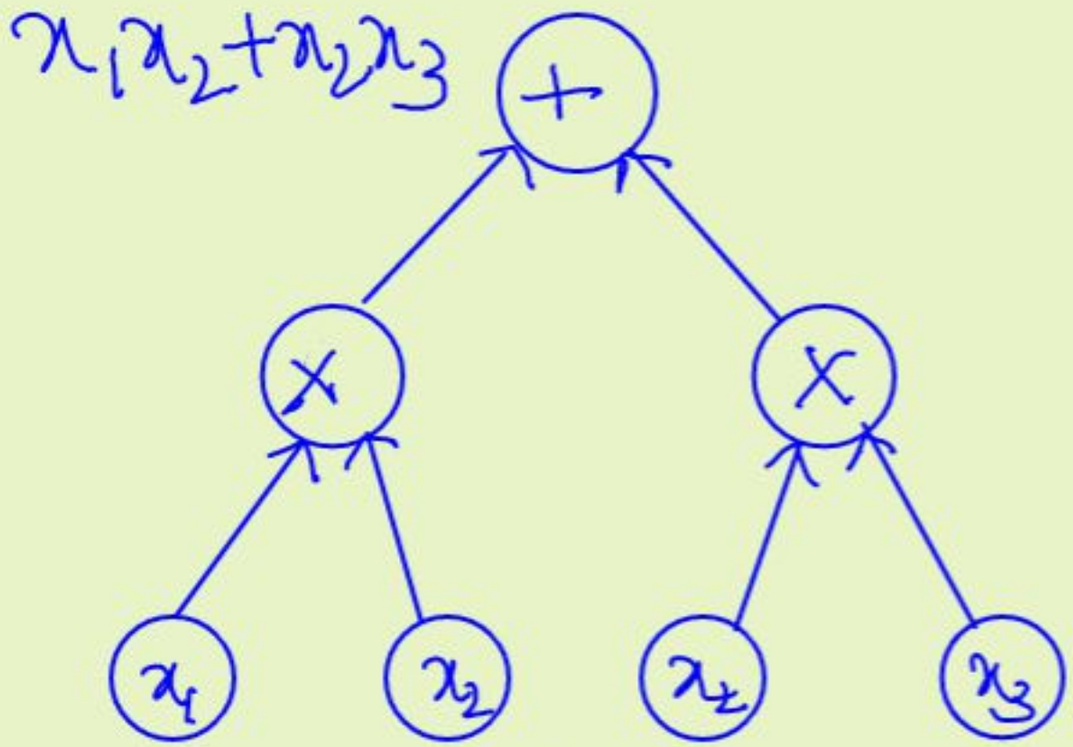
Depth = length of longest input-output path.

Algebraic formulas



→ Underlying graph is a tree

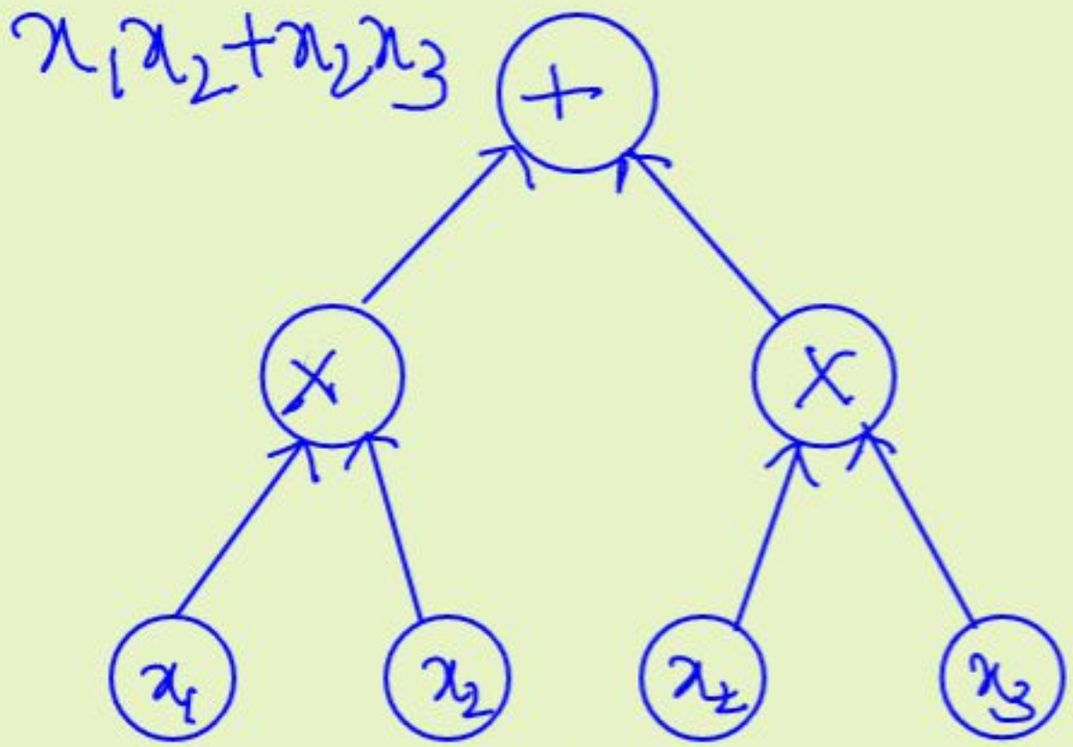
Algebraic formulas



→ Underlying graph is a tree

→ Formulas ↔ algebraic expressions

Algebraic formulas

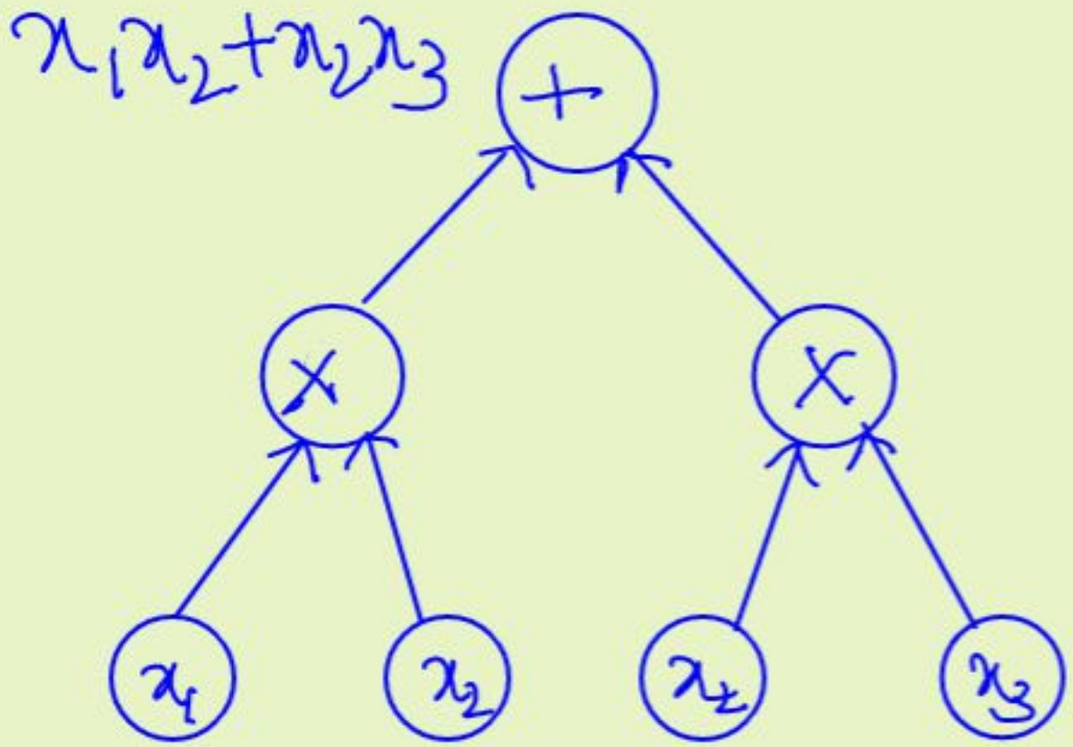


→ Underlying graph is a tree

→ Formulas ↔ algebraic expressions

→ Size of formula ≈ Size of expression

Algebraic formulas



→ Underlying graph is a tree

→ Formulas ⟷ algebraic expressions

→ Size of formula ≈ Size of expression

→ Any $P \in \mathbb{F}[x_1, \ldots, x_n]$ has a "trivial" depth-2 formula of size $\binom{n+d}{d}$

Reductions

→ (P_n) & (Q_m)

Reductions

→ (P_n) & (Q_m)

→ Projection reductions

$$P_n(x_1, \ldots, x_n) = Q_{m(n)}(l_1(\bar{x}), \ldots, l_{m(n)}(\bar{x}))$$

l_i - polynomials of degree ≤ 1

Reductions

→ (P_n) & (Q_m)

→ Projection reductions

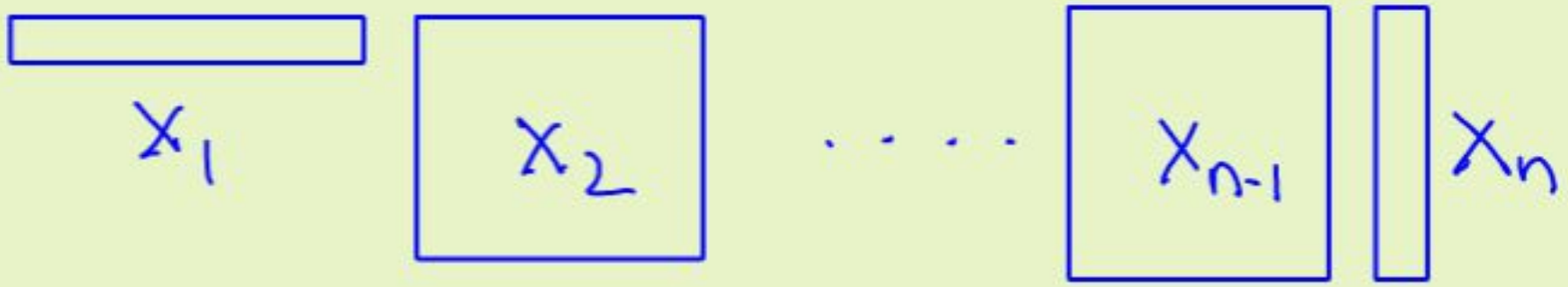
$$P_n(x_1, \ldots, x_n) = Q_{m(n)}(l_1(\bar{x}), \ldots, l_{m(n)}(\bar{x}))$$

l_i - polynomials of degree ≤ 1

→ "Reducing" one computational problem to another.

Reductions to IMM

$IMM_{n,n} =$



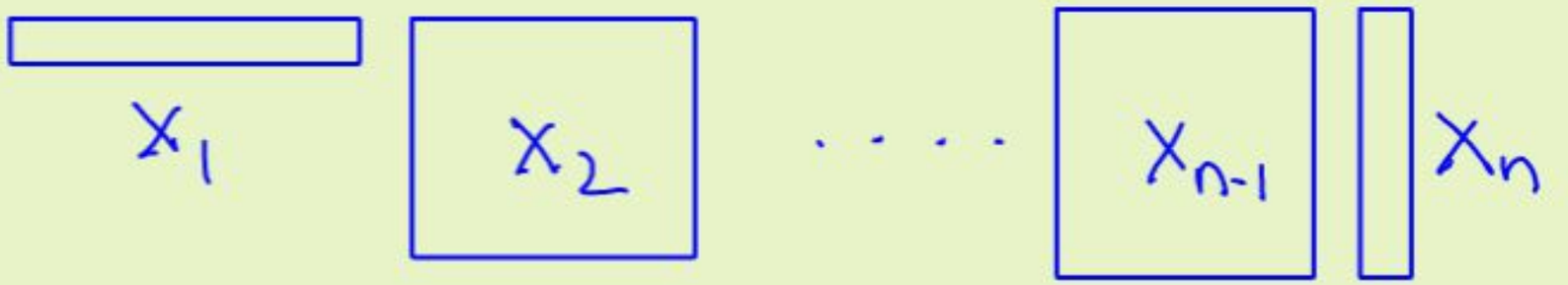
Reductions to IMM

$IMM_{n,n} =$

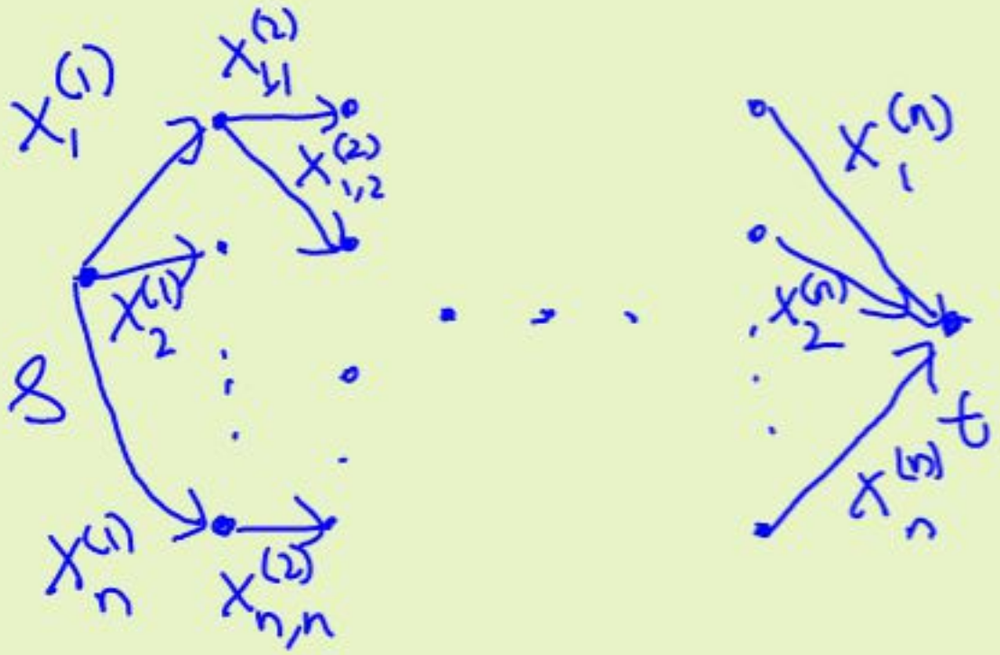
$$= \sum_{i_1, i_2 \cdots, i_{n-1}} X^{(1)}_{i_1} X^{(2)}_{i_1,i_2} X^{(3)}_{i_2,i_3} \cdots X^{(n-1)}_{i_{n-1},i_n} \cdot X^{(n)}_{i_n}$$

Reductions to IMM

$IMM_{n,n} =$

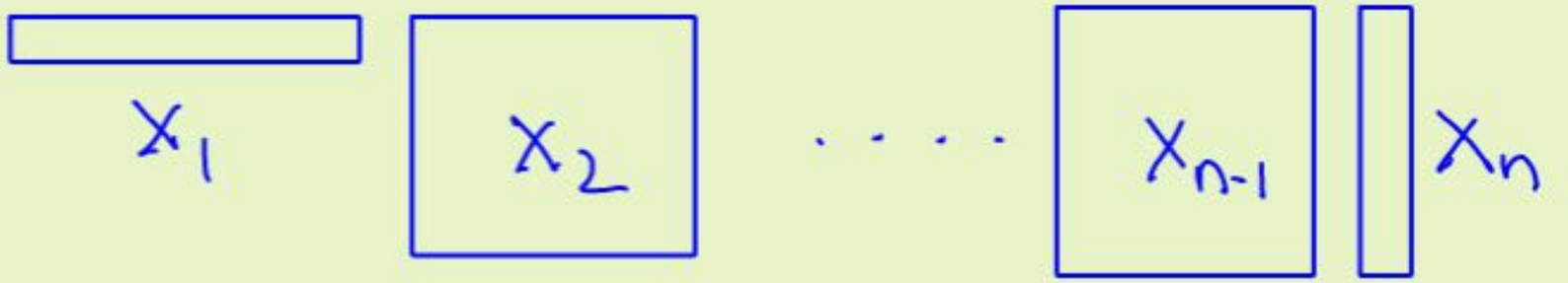


$$= \sum_{i_1, i_2 \cdots, i_{n-1}} X^{(1)}_{i_1} X^{(2)}_{i_1,i_2} X^{(3)}_{i_2,i_3} \cdots X^{(n-1)}_{i_{n-1},i_n} \cdot X^{(n)}_{i_n}$$

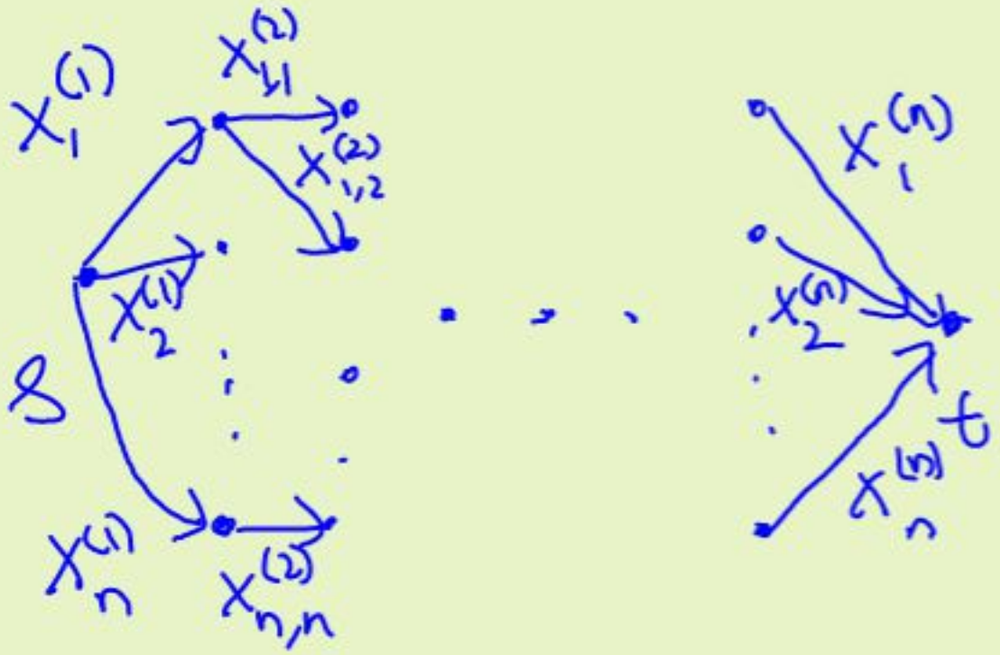


Reductions to IMM

$IMM_{n,n} =$



$$= \sum_{i_1, i_2 \cdots, i_{n-1}} X^{(1)}_{i_1} X^{(2)}_{i_1,i_2} X^{(3)}_{i_2,i_3} \cdots X^{(n-1)}_{i_{n-1},i_n} \cdot X^{(n)}_{i_n}$$



$IMM_{n,n} =$

$$\sum_{s\text{-}t \text{ paths } p} \prod_{\text{edges } e \text{ of } p} \text{label}(e)$$

Algebraic Branching Programs (ABPs)

→ Layered DAG

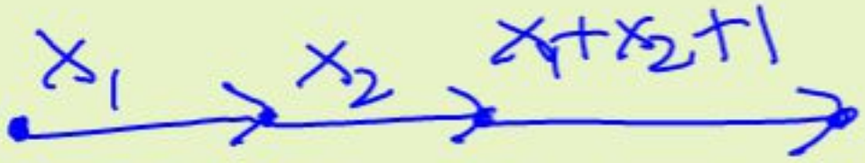
source s &

sink t

→ Edges labelled by linear polys

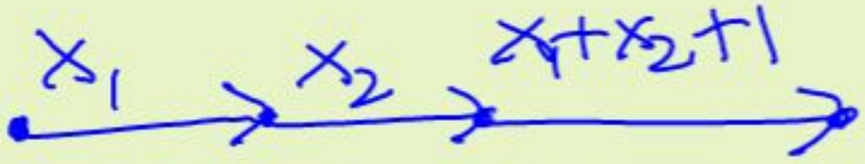
$$P_G = \sum_{\substack{p : s \to t \\ \text{path}}} \prod_{e \text{ on } p} l_e$$

Examples of ABPs

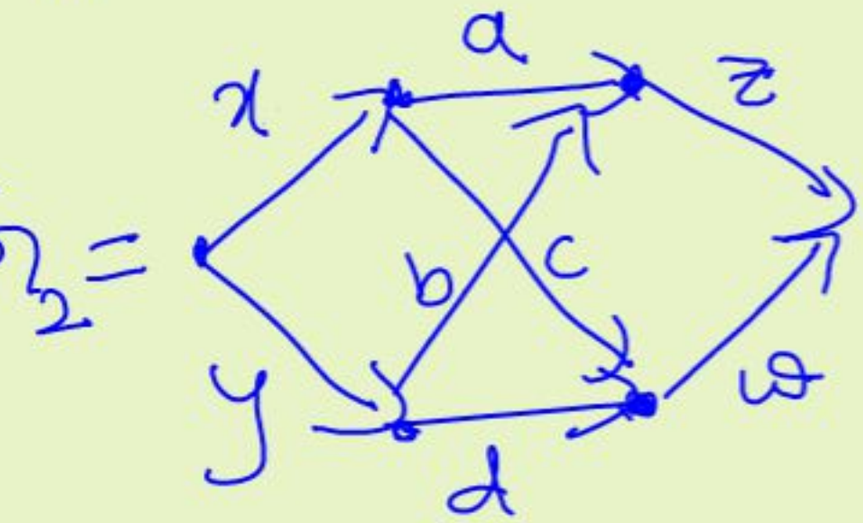
$G_1 =$ 

$P_{G_1} = x_1 x_2 (x_1+x_2+1)$.

Examples of ABPs

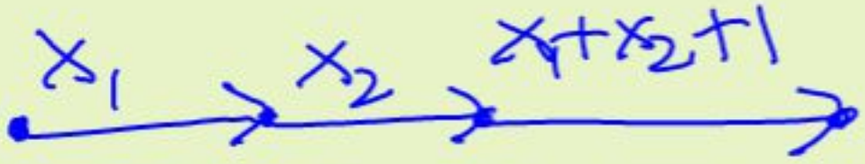
$G_1 =$ 

$P_{G_1} = x_1 x_2 (x_1 + x_2 + 1)$

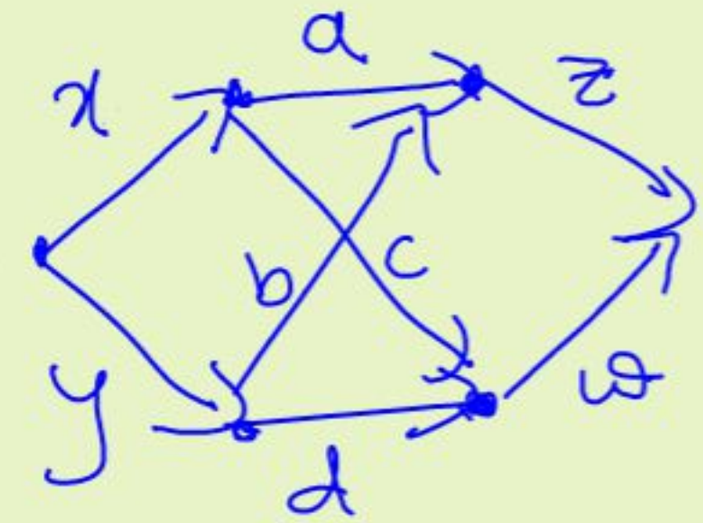
$G_2 =$ 

$P_{G_2} = xaz + xcw + ybz + ydw$

Examples of ABPs

$G_1 =$ 

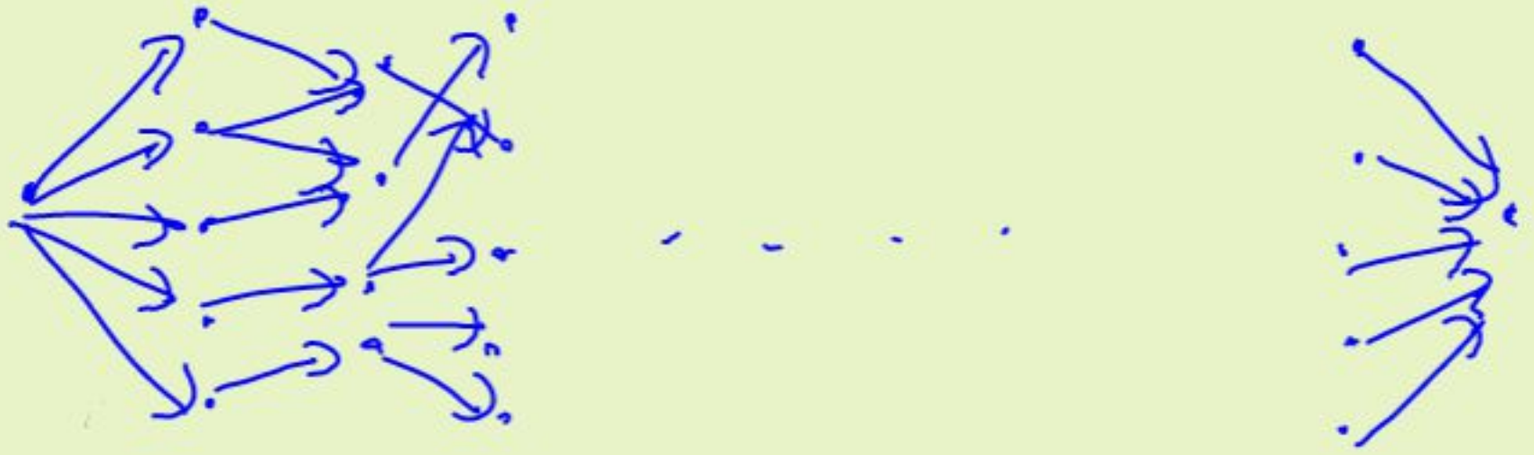
$P_{G_1} = x_1 x_2 (x_1 + x_2 + 1)$

$G_2 =$ 

$P_{G_2} = \lambda a z + x c w + y b z + y d w$

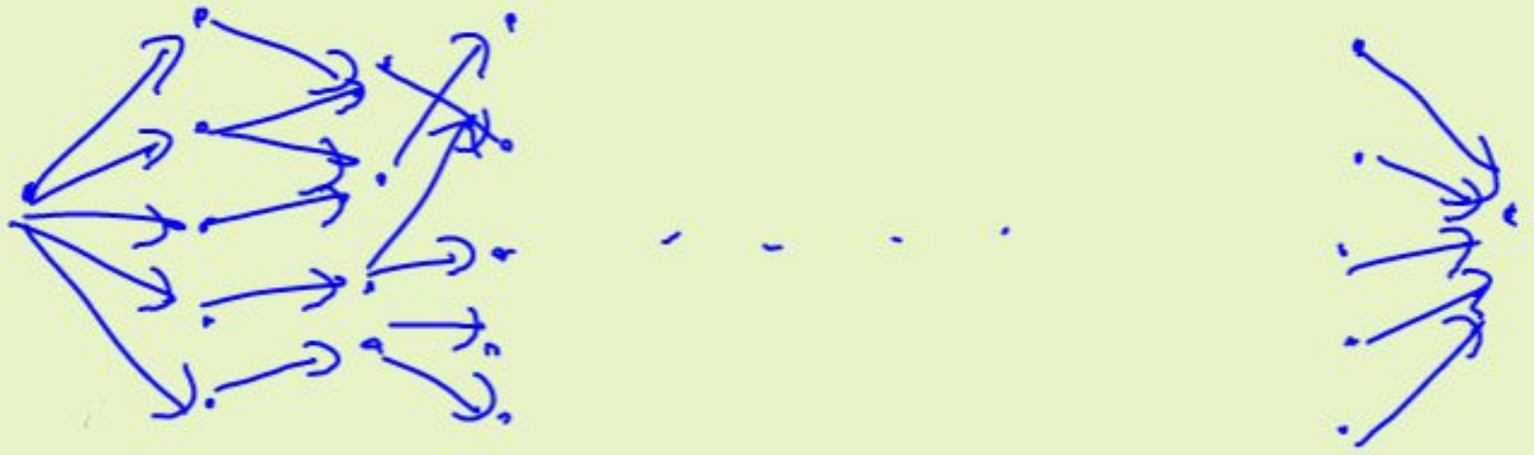
$P_{G_2} = (x, y) \begin{pmatrix} a & c \\ b & d \end{pmatrix} \begin{pmatrix} z \\ w \end{pmatrix}$

ABP size



size = # of vertices in graph

ABP size



size = # of vertices in graph

Obs: ABP A of size s $\Longrightarrow$

P_A a projection of $IMM_{s,s}$

Complexity classes

$(P_n(x_1, \ldots, x_n))_{n \geq 1}$ – a family of polys.

Ckt-size(P_n) = smallest size of a circuit for P_n

Formula-size(P_n), ABP-size(P_n) similar.

Complexity classes

$(p_n(x_1, \ldots, x_n))_{n \geq 1}$ – a family of polys.

Ckt-size(p_n) = smallest size of a circuit for p_n

Formula-size(p_n), ABP-size(p_n) similar.

$(p_n)_{n \geq 1}$ has polynomial-sized circuits if Ckt-size$(p_n) \leq n^{O(1)}$.

Complexity classes

$(p_n(x_1, \ldots, x_n))_{n \geq 1}$ - a family of polys.

VF - polynomial families that have poly-sized formulas

Complexity classes

$(p_n(x_1, \ldots, x_n))_{n \geq 1}$ - a family of polys.

VF - polynomial families that have poly-sized formulas

VBP - poly-sized ABPs

VP - poly-sized circuits

Example 1 : $E_n^d(x_1, \ldots, x_n) = \sum_{|S|=d} \prod_{i \in S} x_i$

Example 1 : $E_n^d(x_1, \ldots, x_n) = \sum_{|S|=d} \prod_{i \in S} x_i$

$$E_n^d = x_n \cdot E_{n-1}^{d-1} + E_{n-1}^d$$

Example 1 : $E_n^d(x_1, \ldots, x_n) = \sum_{|S|=d} \prod_{i \in S} x_i$

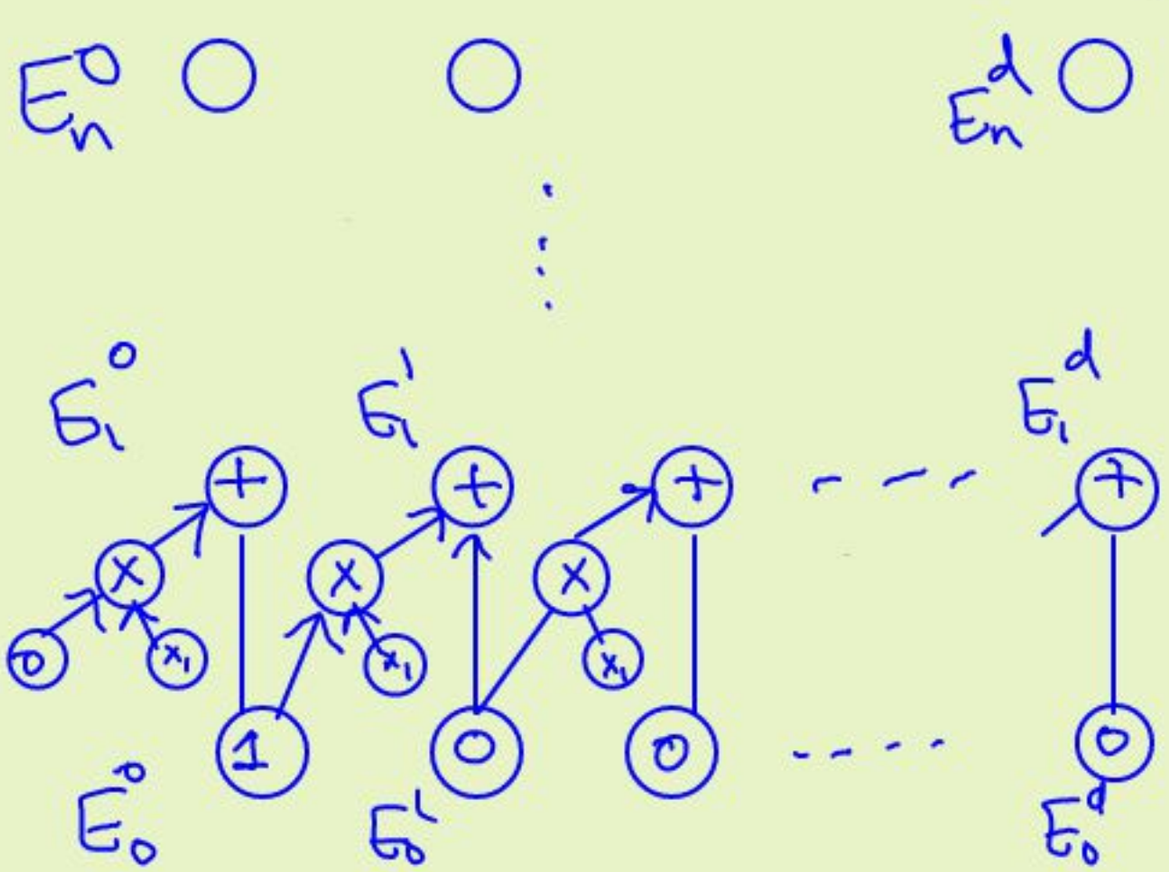
$E_n^d = x_n \cdot E_{n-1}^{d-1} + E_{n-1}^d$

Ckt C for E_n^d: Construct ckts for $E_n^0, \ldots, E_n^d$ inductively

Example 1 : $E_n^d(x_1, \ldots, x_n) = \sum_{|S|=d} \prod_{i \in S} x_i$

$$E_n^d = x_n \cdot E_{n-1}^{d-1} + E_{n-1}^d$$

Ckt C for E_n^d: Construct ckts for $E_n^0, \ldots, E_n^d$ inductively

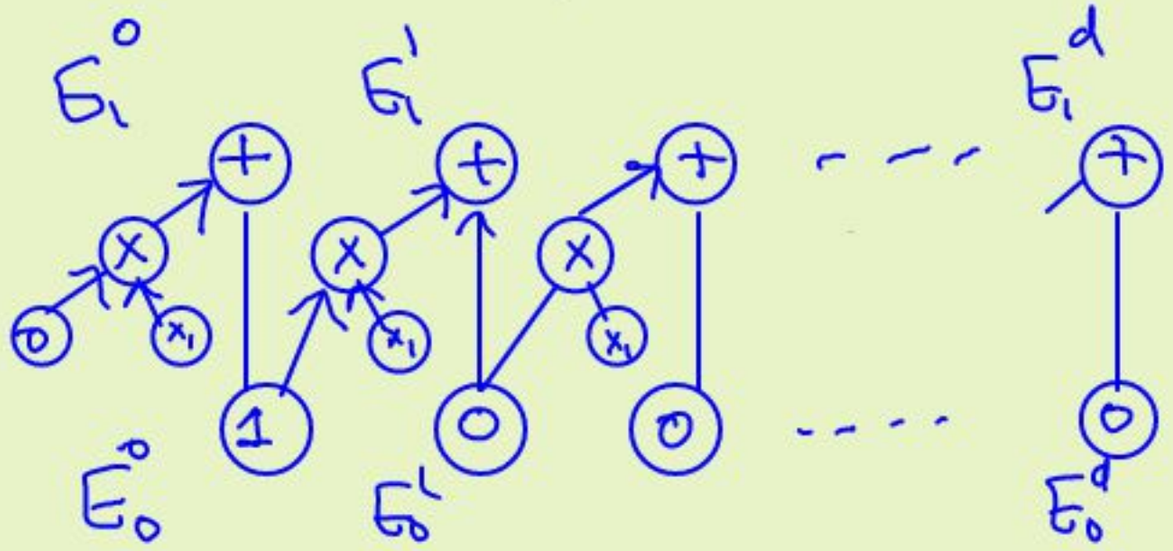


Example 1 : $E_n^d(x_1, \ldots, x_n) = \sum_{|S|=d} \prod_{i \in S} x_i$

$$E_n^d = x_n \cdot E_{n-1}^{d-1} + E_{n-1}^d$$

Ckt C for E_n^d: Construct ckts for $E_n^0, \ldots, E_n^d$ inductively

Size $\leq 4nd$
$= O(n^2)$

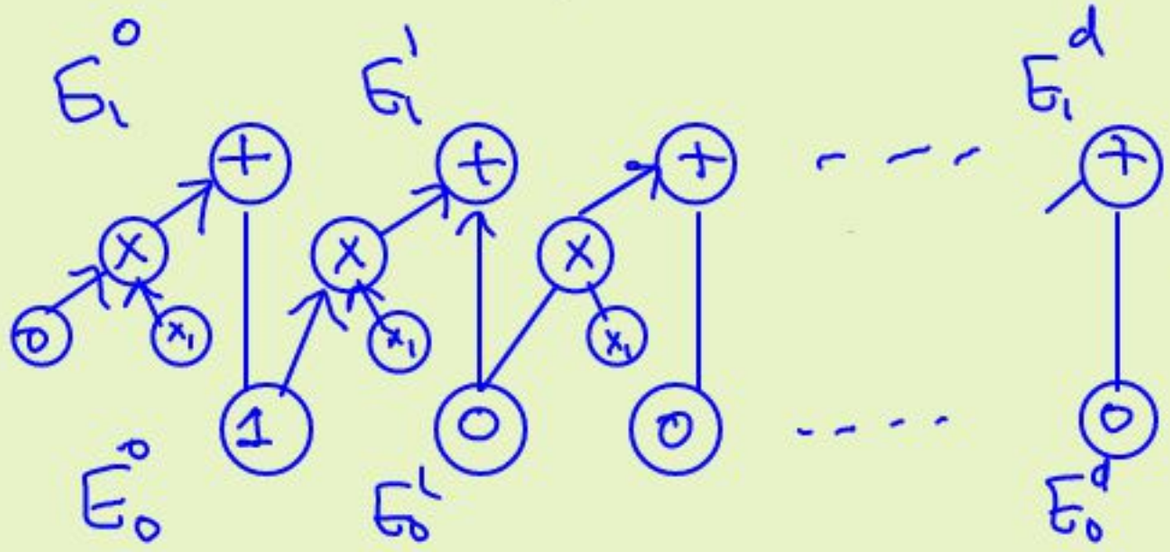


Example 1 : $E_n^d(x_1, \ldots, x_n) = \sum_{|S|=d} \prod_{i \in S} x_i$

$E_n^d = x_n \cdot E_{n-1}^{d-1} + E_{n-1}^d$

Ckt C for E_n^d: Construct ckts for $E_n^0, \ldots, E_n^d$ inductively

Size $\leq 4nd$
$= O(n^2)$



Skew circuit
- every $\times$-gate has an variable or constant input.

Example 1 : $E_n^d(x_1, \ldots, x_n) = \sum_{|S|=d} \prod_{i \in S} x_i$

$$E_n^d = x_n \cdot E_{n-1}^{d-1} + E_{n-1}^d$$

Example 1 : $E_n^d(x_1,\dots,x_n) = \sum_{|S|=d} \prod_{i\in S} x_i$

$$E_n^d = x_n \cdot E_{n-1}^{d-1} + E_{n-1}^d$$

$$(E_n^d, E_n^{d-1}, \dots, E_n^0) = (E_{n-1}^d, \dots, E_{n-1}^0) \begin{pmatrix} 1 & & & & \\ x_n & 1 & & & \\ & x_n & \ddots & & \\ & & \ddots & 1 & \\ & & & x_n & 1 \end{pmatrix}$$

Example 1 : $E_n^d(x_1,\dots,x_n) = \sum_{|S|=d} \prod_{i\in S} x_i$

$$E_n^d = x_n \cdot E_{n-1}^{d-1} + E_{n-1}^d$$

$$(E_n^d, E_n^{d-1}, \dots, E_n^0) = (E_{n-1}^d, \dots, E_{n-1}^0) \begin{pmatrix} 1 & & & & \\ x_n & 1 & & & \\ & x_n & \ddots & & \\ & & \ddots & 1 & \\ & & & x_n & 1 \end{pmatrix}$$

$$= (0,0,\dots,0,1) \begin{pmatrix} 1 & & & \\ x_1 & 1 & & \\ & x_1 & \ddots & \\ & & & x_1 1 \end{pmatrix} \begin{pmatrix} 1 & & \\ x_2 & 1 & \\ & x_2 & \ddots \\ & & x_2 1 \end{pmatrix} \cdots \begin{pmatrix} 1 & & \\ x_n & \ddots & \\ & \ddots & x_n 1 \end{pmatrix}$$

Example 1 : $E_n^d(x_1,\dots,x_n) = \sum_{|S|=d} \prod_{i\in S} x_i$

$$E_n^d = x_n \cdot E_{n-1}^{d-1} + E_{n-1}^d$$

$$(E_n^d, E_n^{d-1}, \dots, E_n^0) = (E_{n-1}^d, \dots, E_{n-1}^0) \begin{pmatrix} 1 & & & & \\ x_n & 1 & & & \\ & x_n & \ddots & & \\ & & \ddots & 1 & \\ & & & x_n & 1 \end{pmatrix}$$

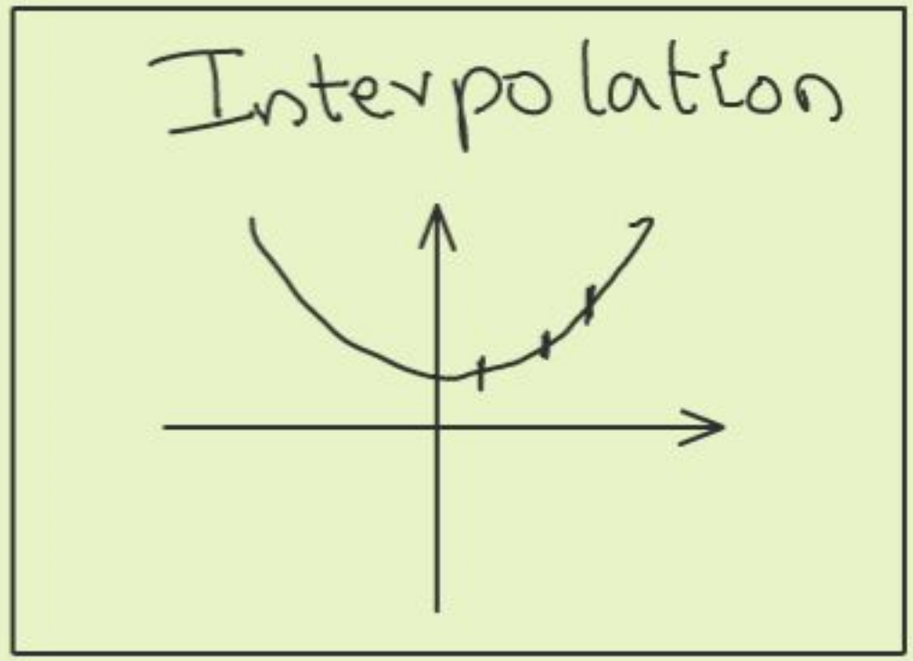
$$= (0,0,\dots,0,1) \begin{pmatrix} 1 & & & \\ x_1 & 1 & & \\ & x_1 & \ddots & \\ & & \ddots & 1 \\ & & & x_1 \end{pmatrix} \begin{pmatrix} 1 & & & \\ x_2 & 1 & & \\ & x_2 & \ddots & \\ & & \ddots & 1 \\ & & & x_2 \end{pmatrix} \cdots \begin{pmatrix} 1 & & & \\ x_n & 1 & & \\ & x_n & \ddots & \\ & & \ddots & 1 \\ & & & x_n \end{pmatrix}$$

Exercise: Draw an ABP for E_n^d!

Example 1 : $E_n^d(x_1, \ldots, x_n) = \underbrace{\sum_{|S|=d} \prod_{i \in S} x_i}_{\binom{n}{d} \text{ terms}}$

Example 1 : $E_n^d(x_1,\ldots,x_n) = \underbrace{\sum_{|S|=d} \prod_{i\in S} x_i}_{\binom{n}{d} \text{ terms}}$

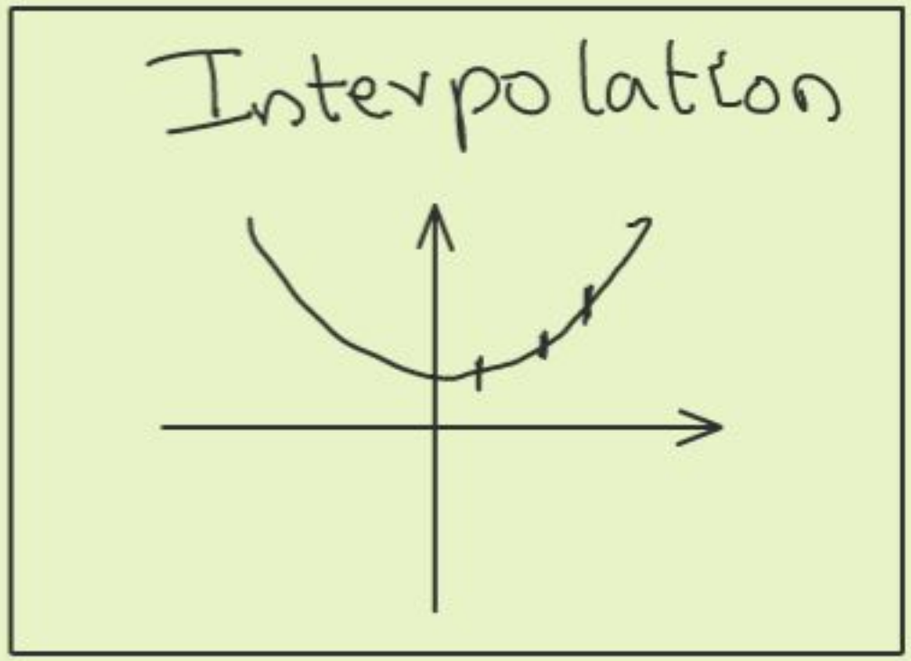
Thm: $E_n^d \in VF$
[Ben-Or]



Example 1: $E_n^d(x_1,\ldots,x_n) = \underbrace{\sum_{|S|=d} \prod_{i\in S} x_i}_{\binom{n}{d} \text{ terms}}$

Thm: $E_n^d \in VF$
[Ben-Or]

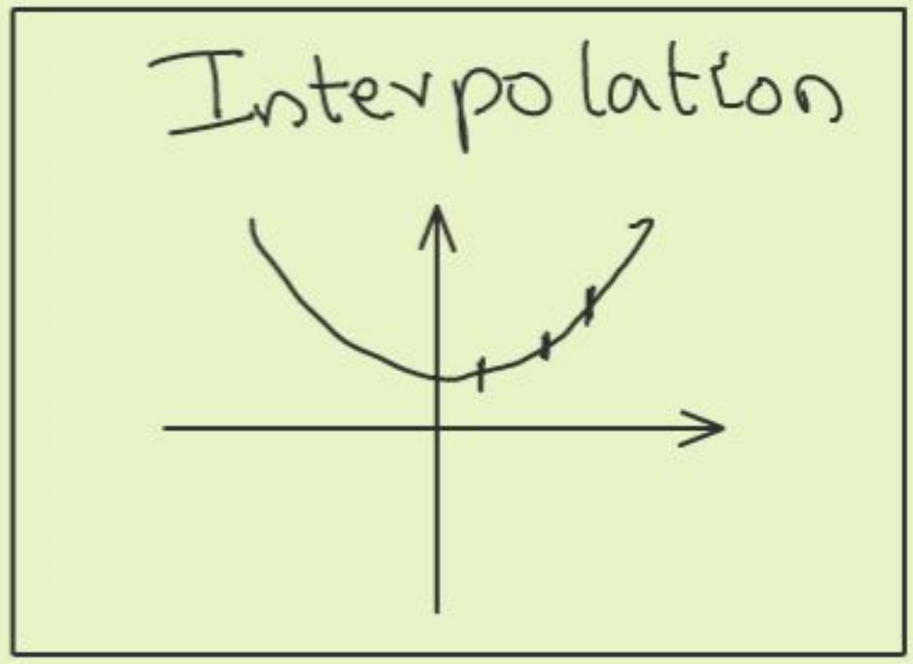
Pf: $\sum_{j=0}^{n} E_n^j \cdot t^j =$



Example 1: $E_n^d(x_1,\dots,x_n) = \underbrace{\sum_{|S|=d} \prod_{i\in S} x_i}_{\binom{n}{d} \text{ terms}}$

Thm: $E_n^d \in VF$
[Ben-Or]

Pf: $\sum_{j=0}^{n} E_n^j \cdot t^j = \prod_{i=1}^{n} (1 + t x_i) = Q(t)$

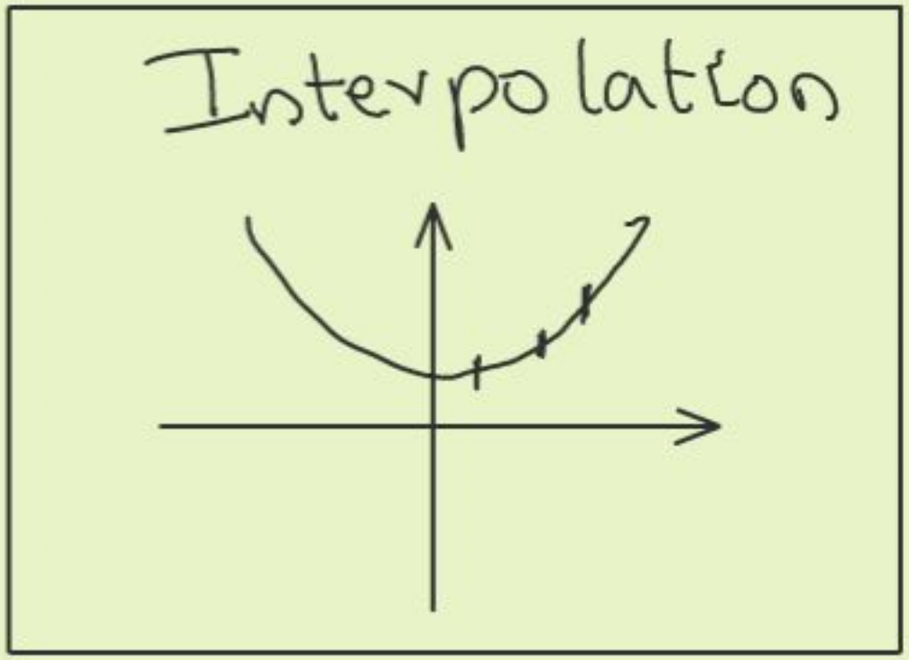


Example 1: $E_n^d(x_1,\dots,x_n) = \underbrace{\sum_{|S|=d} \prod_{i\in S} x_i}_{\binom{n}{d} \text{ terms}}$

Thm: $E_n^d \in VF$ [Ben-Or]

Pf: $\sum_{j=0}^{n} E_n^j \cdot t^j = \prod_{i=1}^{n} (1+tx_i) = Q(t)$

↳ formula of size $O(n)$!



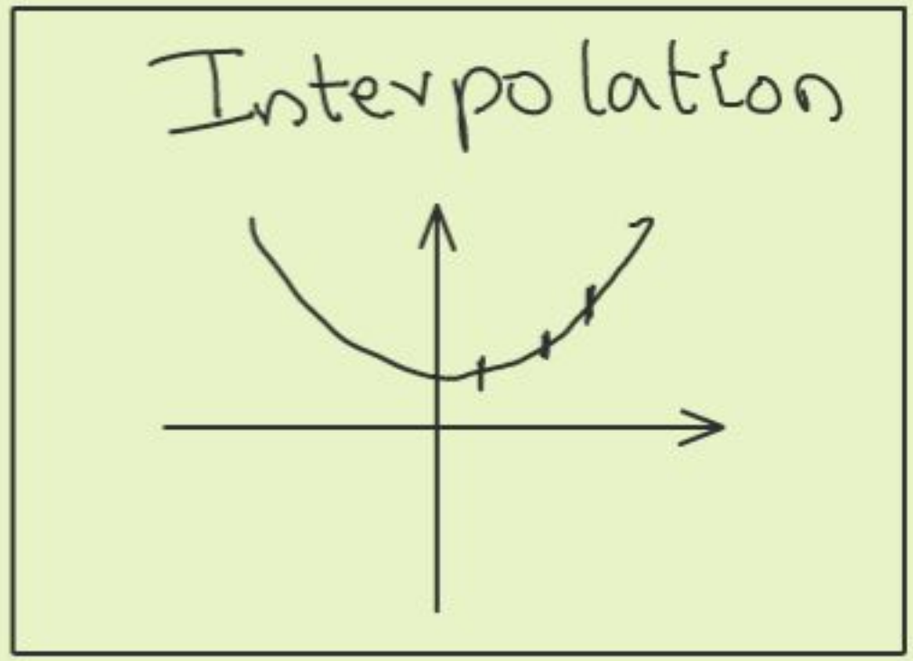
Example 1 : $E_n^d(x_1, \ldots, x_n) = \underbrace{\sum_{|S|=d} \prod_{i \in S} x_i}_{\binom{n}{d} \text{ terms}}$

Thm: $E_n^d \in VF$ [Ben-Or]

Pf: $\sum_{j=0}^{n} E_n^j \cdot t^j = \prod_{i=1}^{n} (1 + t x_i) = Q(t)$

↳ formula of size $O(n)$!

E_n^d = co-eff. of t^d in Q



Example 1 : $E_n^d(x_1, \ldots, x_n) = \underbrace{\sum_{|S|=d} \prod_{i \in S} x_i}_{\binom{n}{d} \text{ terms}}$

Thm: $E_n^d \in VF$
[Ben-Or]

Pf: $\sum_{j=0}^{n} E_n^j \cdot t^j = \prod_{i=1}^{n} (1 + t x_i) = Q(t)$

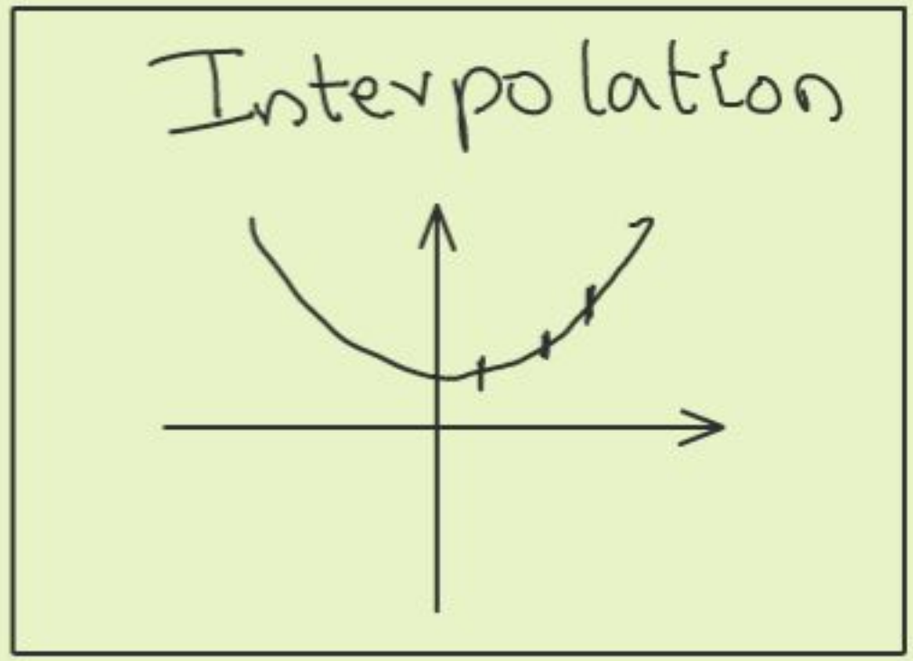
↳ formula of size $O(n)$!

E_n^d = co-eff. of t^d in Q

$= \sum_{j=0}^{n} \beta_j Q(\alpha_j)$

$\alpha_0, \ldots, \alpha_n \in F$ distinct

$\beta_0, \ldots, \beta_n \in F$



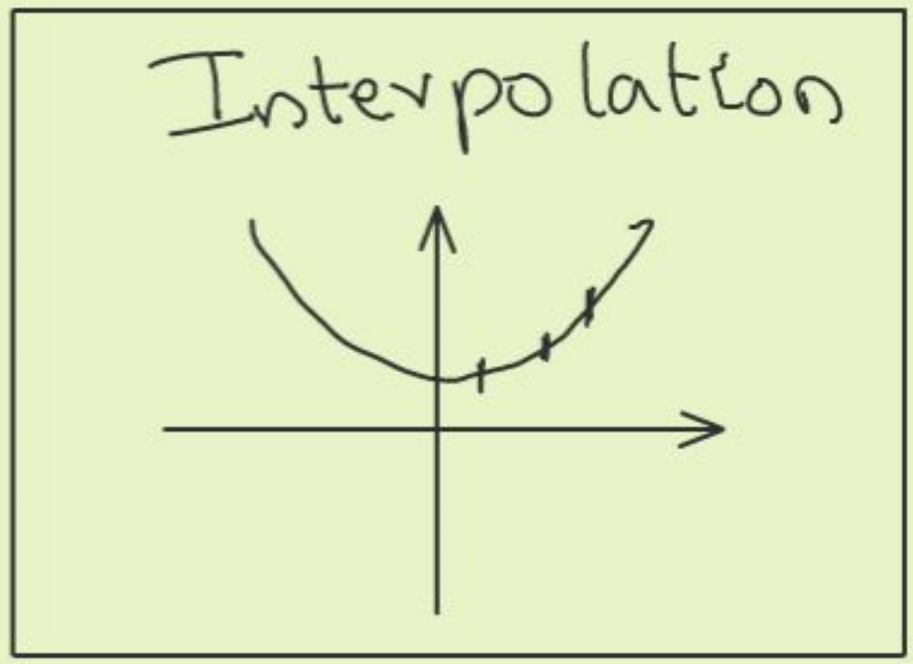
Example 1: $E_n^d(x_1,\dots,x_n) = \underbrace{\sum_{|S|=d} \prod_{i\in S} x_i}_{\binom{n}{d} \text{ terms}}$

Thm: $E_n^d \in VF$ [Ben-Or]

Pf:
$$\sum_{j=0}^{n} E_n^j \cdot t^j = \prod_{i=1}^{n} (1+tx_i) = Q(t)$$

↳ formula of size $O(n)$!

$$E_n^d = \sum_{j=0}^{n} \beta_j \prod_{i=1}^{n} (1+\alpha_j x_i)$$



Complexity classes

$(p_n(x_1, \ldots, x_n))_{n \geq 1}$ - a family of polys.

VF - polynomial families that have poly-sized formulas

VBP - poly-sized ABPs

VP - poly-sized circuits

Complexity classes

$(p_n(x_1, \ldots, x_n))_{n \geq 1}$ - a family of polys.

VF - polynomial families that have poly-sized formulas

VBP - poly-sized ABPs

VP - poly-sized circuits

$$VF \subseteq VBP \subseteq VP$$

Complexity classes

$(p_n(x_1, \ldots, x_n))_{n \geq 1}$ - a family of polys.

VF - polynomial families that have poly-sized formulas

VBP - poly-sized ABPs

VP - poly-sized circuits

$$VF \subseteq VBP \subseteq VP$$

strict?

strict?

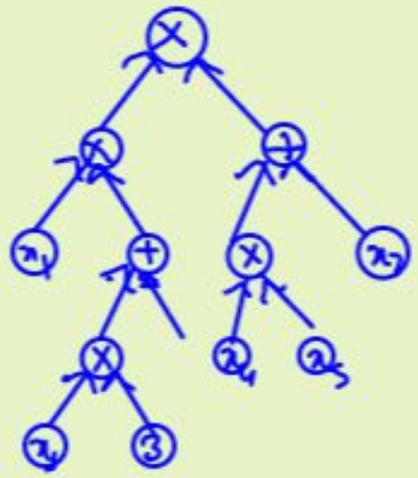
$VF \subseteq VBP$

Thm: Formula F of size s can be made ABP A of size $O(s)$.

VF ⊆ VBP

Thm: Formula F of size s can be made ABP A of size O(s).

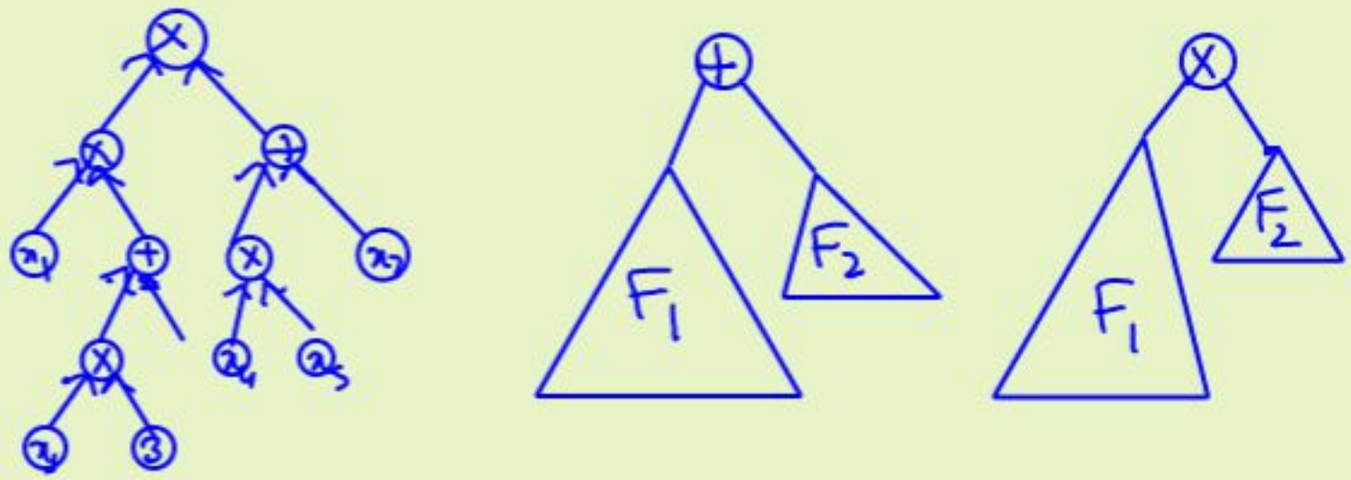
Formula-



VF ⊆ VBP

Thm: Formula F of size s can be made ABP A of size $O(s)$.

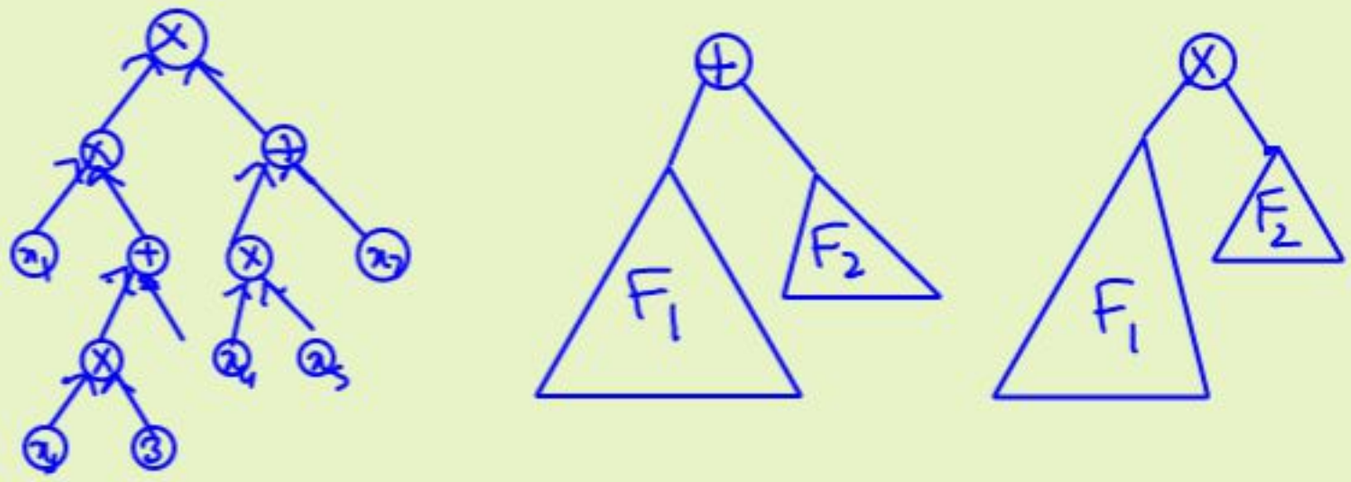
Formula-



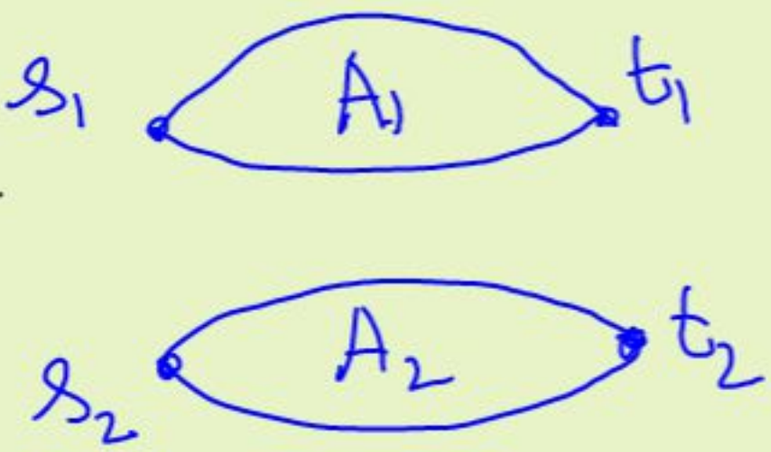
$VF \subseteq VBP$

Thm: Formula F of size s can be made ABP A of size $O(s)$.

Formula-



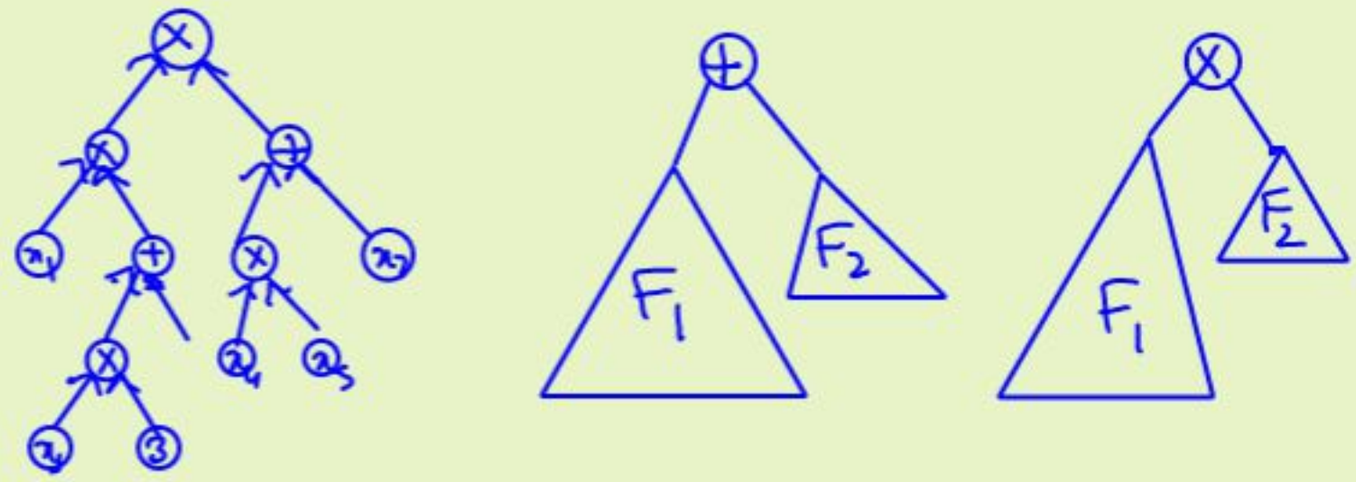
ABP -



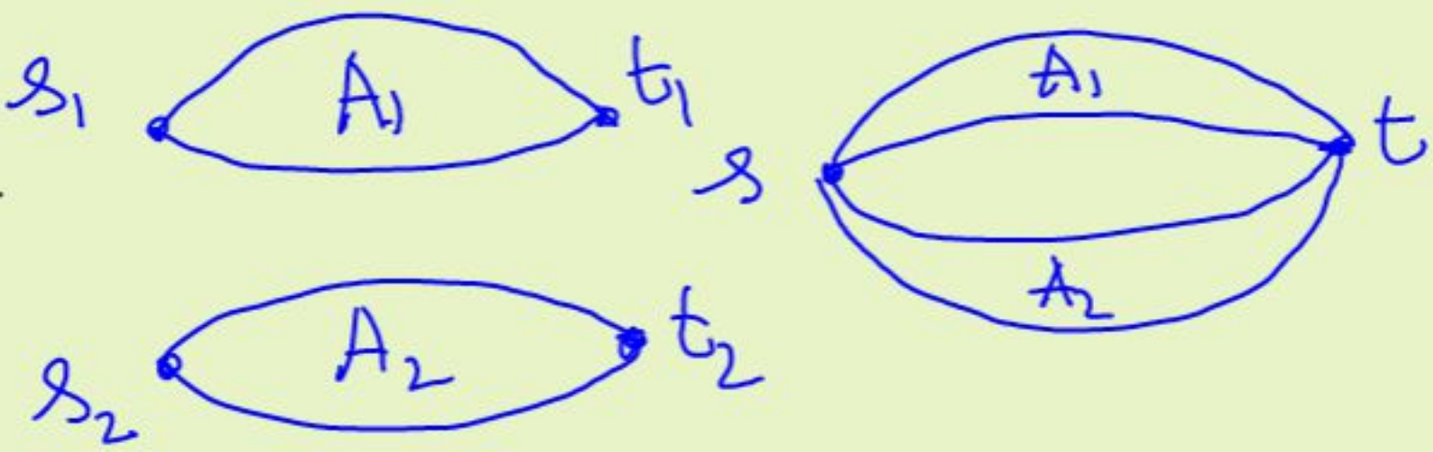
VF ⊆ VBP

Thm: Formula F of size s can be made ABP A of size $O(s)$.

Formula-



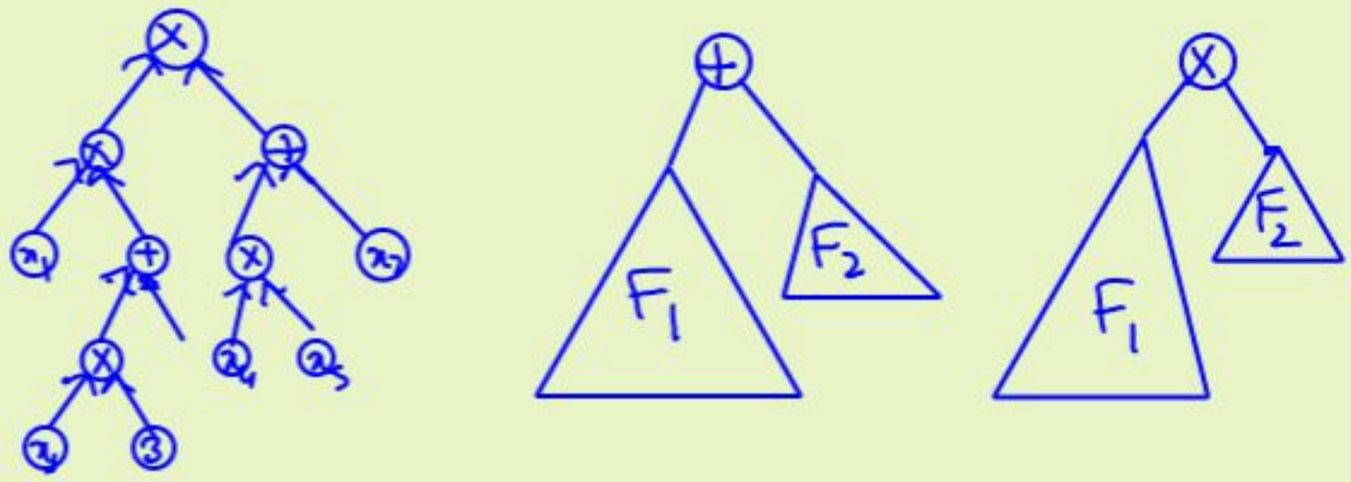
ABP -



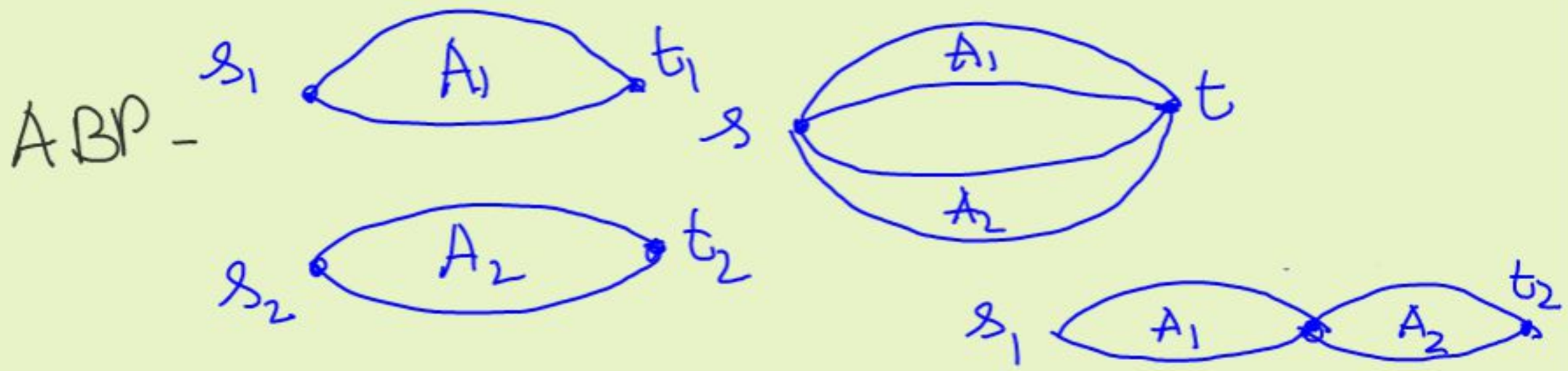
VF ⊆ VBP

Thm: Formula F of size s can be made ABP A of size $O(s)$.

Formula-



ABP -



VBP $\subseteq$ VP

VBP $\subseteq$ VP

Thm: $IMM_{n,n}$ has a circuit of size $O(n^3)$

$VBP \subseteq VP$

Thm: $IMM_{n,n}$ has a circuit of size $O(n^3)$

Pf: Standard Matrix Mult. algorithm.

$VBP \subseteq VP$

Thm: $IMM_{n,n}$ has a circuit of size $O(n^3)$

Pf: Standard Matrix Mult. algorithm.

Cor: ABP A of size $s \Rightarrow$ Ckt. C of size $O(s^3 n)$
& n variables

$VBP \subseteq VP$

Thm: $IMM_{n,n}$ has a circuit of size $O(n^3)$

Pf: Standard Matrix Mult. algorithm.

Cor: ABP A of size $s \Rightarrow$ Ckt. C of size $O(s^3 n)$
& n variables

$\rightarrow P_A(x_1, \ldots, x_n)$ is a projection of $IMM_{s,s}$

$VBP \subseteq VP$

Thm: $IMM_{n,n}$ has a circuit of size $O(n^3)$

Pf: Standard Matrix Mult. algorithm.

Cor: ABP A of size $s \Rightarrow$ Ckt. C of size $O(s^3 n)$
& n variables

$\rightarrow P_A(x_1, \ldots, x_n)$ is a projection of $IMM_{s,s}$

$\rightarrow IMM_{s,s}$ has a ckt. C' of size $O(s^3)$.

$VBP \subseteq VP$

Thm: $IMM_{n,n}$ has a circuit of size $O(n^3)$

Pf: Standard Matrix Mult. algorithm.

Cor: ABP A of size $s \Rightarrow$ Ckt. C of size $O(s^3 n)$
& n variables

→ $P_A(x_1, \ldots, x_n)$ is a projection of $IMM_{s,s}$

→ $IMM_{s,s}$ has a ckt. C' of size $O(s^3)$.

→ linear substitution in C' gives C.

Efficient Reductions

Efficient Reductions

→ (P_n) & (Q_m)

→ Projection reductions

$$P_n(x_1, \ldots, x_n) = Q_{m(n)}(l_1(\bar{x}), \ldots, l_{m(n)}(\bar{x}))$$

l_i - polynomials of degree ≤ 1

Efficient Reductions

→ (P_n) & (Q_m)

→ Projection reductions

$$P_n(x_1, \ldots, x_n) = Q_{m(n)}(\ell_1(\bar{x}), \ldots, \ell_{m(n)}(\bar{x}))$$

ℓ_i - polynomials of degree ≤ 1

→ p-Projections: $m(n) \leq n^{O(1)}$

→ VE, VBP, VP closed under p-projections.

Example 2 : $\det_n$

Example 2 : $\det_n$

Thm : $\det_n \in VP$

→ Gaussian elimination (uses divisions) [Ex!]

Example 2 : $\det_n$

Thm : $\det_n \in VP$

→ Gaussian elimination (uses divisions) [Ex!]

Thm [Strassen] : P has a circuit of size s with divisions $\Rightarrow$ P has a circuit of size $(s \cdot \deg(P))^{O(1)}$.

Example 2 : $\det_n$

Thm : $\det_n \in VP$

→ Gaussian elimination (uses divisions) [Ex!]

Thm [Strassen] : P has a circuit of size s with divisions $\Rightarrow$ P has a circuit of size $(s \cdot \deg(P))^{O(1)}$.

Thm [Berkowitz] : $\det_n \in VBP$.

Cor : $\det_n$ a p-projection of $IMM_{n,n}$.

Example 2 : det_n

Thm : $det_n \in VP$

→ Gaussian elimination (uses divisions) [Ex!]

Thm [Strassen] : P has a circuit of size s with divisions $\Rightarrow$ P has a circuit of size $(s \cdot deg(P))^{O(1)}$.

Thm [Berkowitz] : $det_n \in VBP$.

Cor : det_n a p-projection of $IMM_{n,n}$.

Thm: [Damm, Vinay, Toda] : $IMM_{n,n}$ p-projection of det_n.

Example 2 : $\det_n$

Thm : $\det_n \in VP$

→ Gaussian elimination (uses divisions) [Ex!]

Thm [Strassen] : P has a circuit of size s with divisions $\Rightarrow$ P has a circuit of size $(s \cdot \deg(P))^{O(1)}$.

Thm [Berkowitz] : $\det_n \in VBP$.

Cor : $\det_n$ a p-projection of $IMM_{n,n}$.

Thm: [Damm, Vinay, Toda] : $IMM_{n,n}$ p-projection of $\det_n$.

Cor : $(P_m)_m \in VBP \Rightarrow (P_n)$ a p-projection of $(\det_n)$.

Completeness

$\mathcal{C}$ - class of polynomial sequences

$(P_n)_{n\geq 1}$ $\mathcal{C}$-complete if

$\rightarrow (P_n)_{n\geq 1} \in \mathcal{C}$

$\rightarrow$ Each $(Q_m)_{m\geq 1} \in \mathcal{C}$ is a p-projection of $(P_n)_{n\geq 1}$

Completeness

$\mathcal{C}$ - class of polynomial sequences

$(P_n)_{n\geq 1}$ $\mathcal{C}$-complete if

$\rightarrow (P_n)_{n\geq 1} \in \mathcal{C}$

$\rightarrow$ Each $(Q_m)_{m\geq 1} \in \mathcal{C}$ is a p-projection of $(P_n)_{n\geq 1}$

Eg : $IMM_{n,n}$ & det_n are VBP-complete.

$VF = VBP$ iff $IMM_{n,n} \in VF$ iff $det_n \in VF$

Lower bounds question

Are there polynomials that do not have small formulas/ABPs/ckts?

Lower bounds question

Are there polynomials that do not have small formulas/ABPs/ckts?

YES. By "counting" arguments

Lower bounds question

Are there polynomials that do not have small formulas / ABPs / ckts?

YES. By "counting" arguments

Thm [HY]: For any n, d, $\exists p_n \in \mathbb{F}[x_1, \ldots x_n]$

$\rightarrow \deg(p_n) \leq d$, $\rightarrow$ Coeffs$(p) \in \{0,1\}$

$\rightarrow$ Any ckt. for p_n is "large."
[essentially trivial]

Lower bounds question

Are there explicit polynomials that do not have small formulas/ABPs/ckts?

Lower bounds question

Are there explicit polynomials that do not have small formulas / ABPs / ckts?

Explicit – VNP

Lower bounds question

Are there explicit polynomials that do not have small formulas / ABPs / ckts?

Explicit – VNP

Special case – $(p_n(x_1,\dots,x_n))_{n\geq 1}$ explicit if

$n \longrightarrow$ [Poly-time algo] $\longrightarrow$ Coeff$(\bar{x}^{\bar{e}})$ in p_n

$e_1,\dots,e_n \geq 0$

$\sum_j e_j \leq d$

Formal definition of VNP

$(P_n)_{n\geq 1} \in VNP$ if $\exists\ (Q_m)_{m\geq 1} \in VP$ s.t

Formal definition of VNP

$(P_n)_{n\geq 1} \in \text{VNP}$ if $\exists\ (Q_m)_{m\geq 1} \in \text{VP}$ s.t

$$P_n(x_1, \ldots, x_n) = \sum_{y_1, \ldots, y_r \in \{0,1\}} Q_{n+r}(x_1, \ldots, x_n, y_1, \ldots, y_r)$$

where $r(n) = n^{O(1)}$.

Formal definition of VNP

$(P_n)_{n\geq 1} \in VNP$ if $\exists\ (Q_m)_{m\geq 1} \in VP$ s.t

$$P_n(x_1, \ldots, x_n) = \sum_{y_1, \ldots, y_r \in \{0,1\}} Q_{n+r}(x_1, \ldots, x_n, y_1, \ldots, y_r)$$

where $r(n) = n^{O(1)}$.

Obs: $VP \subseteq VNP$.

Formal definition of VNP

$(P_n)_{n\geq 1} \in VNP$ if $\exists\ (Q_m)_{m\geq 1} \in VP$ s.t

$$P_n(x_1,\dots,x_n) = \sum_{y_1,\dots,y_r \in \{0,1\}} Q_{n+r}(x_1,\dots,x_n,y_1,\dots,y_r)$$

where $r(n) = n^{O(1)}$.

Obs: $VP \subseteq VNP$.

Ex: Show $per_m \in VNP$ using above definition

Hint: $r = m^2$ & $Q(x,M) = \begin{cases} 0 & \text{if } M \text{ not a permutation matrix} \\ \sigma^{th} \text{ term in } per_m & M = M_\sigma (\sigma \in S_m) \end{cases}$

Valiant's Hypothesis

→ Is $VP = VNP$? Conjectured: NO.

Valiant's Hypothesis

→ Is $VP = VNP$? Conjectured: NO.

Thm [Valiant]: per_n is VNP-complete.

Cor: $VP = VNP$ iff $per_n \in VP$.

Valiant's Hypothesis

→ Is $VP = VNP$? Conjectured: NO.

Thm [Valiant]: per_n is VNP-complete.

Cor: $VP = VNP$ iff $per_n \in VP$.

Cor: $VNP = VBP$ iff $per_n \in VBP$

iff per_n is a p-projection of det_m / $IMM_{m,m}$

Valiant's Hypothesis

→ Is $VP = VNP$? Conjectured: NO.

Thm [Valiant]: per_n is VNP-complete.

Cor: $VP = VNP$ iff $per_n \in VP$.

Cor: $VNP = VBP$ iff $per_n \in VBP$

iff per_n is a p-projection of $det_m / IMM_{m,m}$

Cor: per_n not a qp-projection of $det_m / IMM_{m,m}$

$n^{O(\log n)} \Rightarrow VP \neq VNP$

Valiant's Hypothesis

→ Is $VP = VNP$? Conjectured: NO.

Thm [Valiant]: per_n is VNP-complete.

Cor: $VP = VNP$ iff $per_n \in VP$.

Cor: $VNP = VBP$ iff $per_n \in VBP$

iff per_n is a p-projection of $det_m / IMM_{m,m}$

Cor: per_n not a qp-projection of $det_m / IMM_{m,m}$

$n^{O(\log n)} \Rightarrow VP \neq VNP$

Known: per_n a projection of det_{2^n}
[Grenet]

Why study VP vs VNP?

1. Formally easier than the (non-uniform) P vs. NP question.

Why study VP vs VNP?

1. Formally easier than the (non-uniform) P vs. NP question.

Thm [Bürgisser]: $P \neq NP \Rightarrow VP \neq VNP$ *

Why study VP vs VNP?

1. Formally easier than the (non-uniform) P vs. NP question.

Thm [Bürgisser]: $P \neq NP \Rightarrow VP \neq VNP$ *

*- for finite fields / $\mathbb{C}$ under GRH.

Why study VP vs VNP?

1. Formally easier than the (non-uniform) P vs. NP question.

Thm [Bürgisser]: $P \neq NP \Rightarrow VP \neq VNP$ *

*- for finite fields / $\mathbb{C}$ under GRH.

2. Polynomial Identity Testing

Polynomial Identity Testing

$$(x_1^2+x_2^2+x_3^2+x_4^2)\cdot(y_1^2+y_2^2+y_3^2+y_4^2) = (x_1y_1-x_2y_2-x_3y_3-x_4y_4)^2 + (x_1y_2+x_2y_1+x_3y_4-x_4y_3)^2 + (x_1y_3-x_2y_4+x_3y_1+x_4y_2)^2 + (x_1y_4+x_2y_3-x_3y_2+x_4y_1)^2$$

Polynomial Identity Testing

$$(x_1^2+x_2^2+x_3^2+x_4^2)\cdot(y_1^2+y_2^2+y_3^2+y_4^2) = (x_1y_1-x_2y_2-x_3y_3-x_4y_4)^2 + (x_1y_2+x_2y_1+x_3y_4-x_4y_3)^2 + (x_1y_3-x_2y_4+x_3y_1+x_4y_2)^2 + (x_1y_4+x_2y_3-x_3y_2+x_4y_1)^2$$

Input: $P(x_1,\ldots,x_n)$ (e.g. as ckt / formula / ABP)

Output: Is $P=0$?

Polynomial Identity Testing

$$(x_1^2+x_2^2+x_3^2+x_4^2)\cdot(y_1^2+y_2^2+y_3^2+y_4^2) = (x_1y_1-x_2y_2-x_3y_3-x_4y_4)^2 + (x_1y_2+x_2y_1+x_3y_4-x_4y_3)^2 + (x_1y_3-x_2y_4+x_3y_1+x_4y_2)^2 + (x_1y_4+x_2y_3-x_3y_2+x_4y_1)^2$$

Input: $P(x_1,\ldots,x_n)$ (e.g. as ckt/formula/ABP)

Output: Is $P=0$?

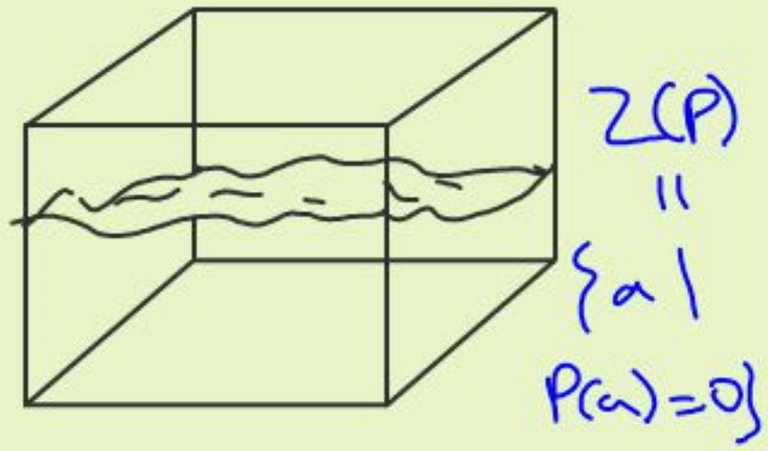
Obvious algm: Expand and check!

Takes time $\binom{n+d}{d}$. Want poly-time.

Randomized Algorithm

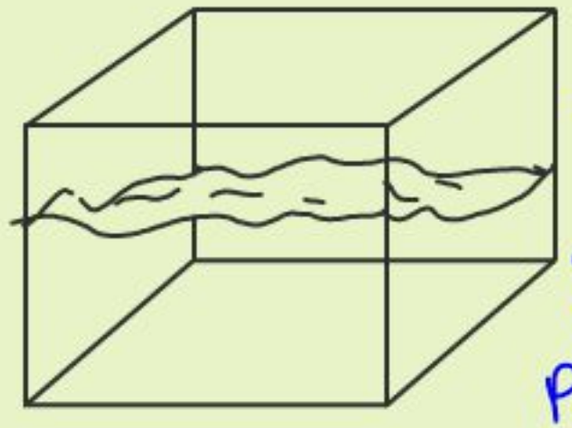
Randomized Algorithm

Non-zero P defines a hyper-surface in $\mathbb{F}^n$. "Most points" not on hypersurface.



Randomized Algorithm

Non-zero P defines a hyper-surface in $\mathbb{F}^n$. "Most points" not on hypersurface.



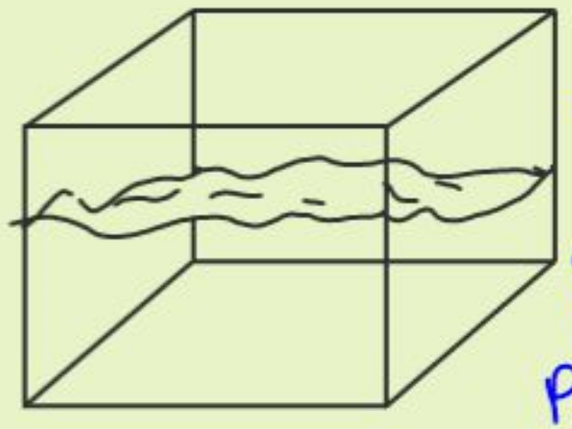
$Z(P)$ = $\{a \mid P(a)=0\}$

Polynomial Identity Lemma [Schwarz, Zippel, DeMillo, Lipton, Ore]

$$\frac{|Z(P) \cap S^n|}{|S|^n} \leq \frac{\deg(P)}{|S|}$$

Randomized Algorithm

Non-zero P defines a hyper-surface in $\mathbb{F}^n$. "Most points" not on hypersurface.



$Z(P)$ = $\{a \mid P(a)=0\}$

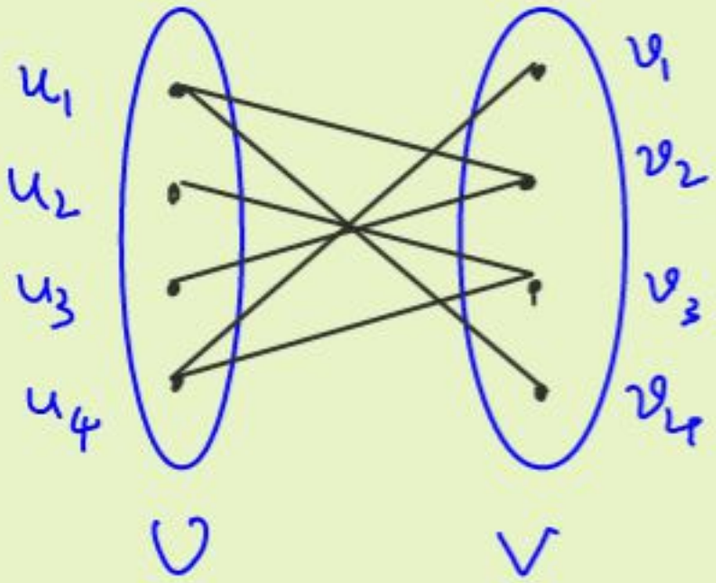
Polynomial Identity Lemma [Schwarz, Zippel, DeMillo, Lipton, Ore]

$$\frac{|Z(P) \cap S^n|}{|S|^n} \leq \frac{\deg(P)}{|S|}$$

Algm. ① Fix $S \subseteq \mathbb{F}$ s.t. $|S| \geq 10d$.

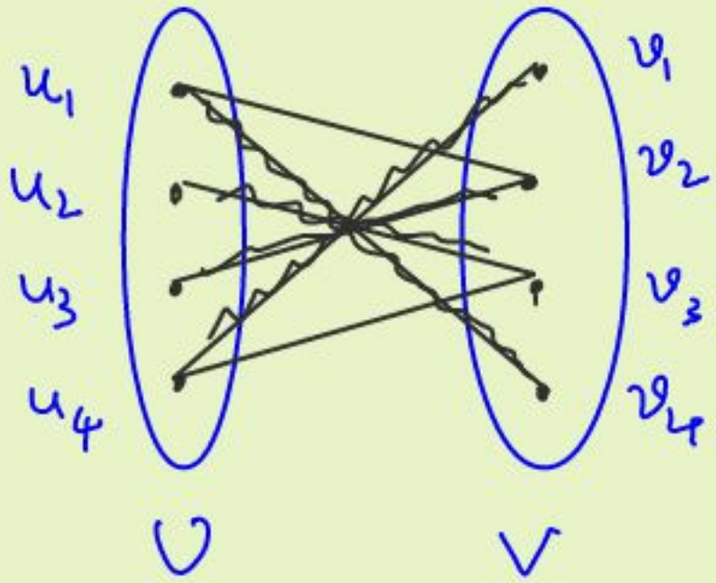
② Choose $a \in_R S^n$ & test if $P(a)=0$.

Matchings



$G = (U, V, E)$, $E \subseteq U \times V$

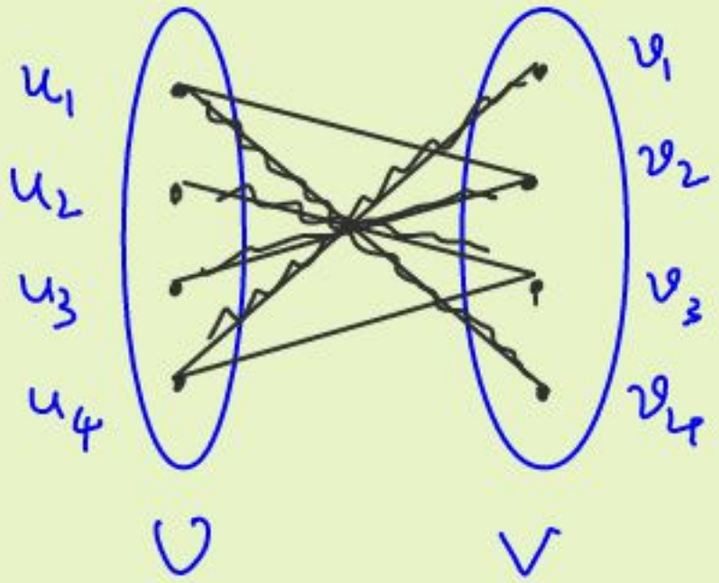
Matchings



$G = (U, V, E)$, $E \subseteq U \times V$

$M \subseteq E$ a Perfect Matching if it defines a 1-1 correspondence b/w U, V.

Matchings



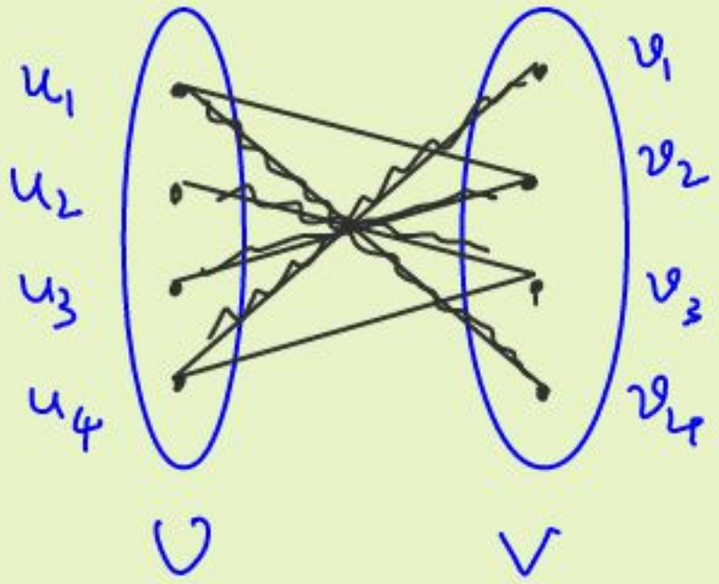
Input: $G=(U,V,E)$

Qn: Does G have a Perfect Matching?

$G=(U,V,E)$, $E \subseteq U \times V$

$M \subseteq E$ a Perfect Matching if it defines a 1-1 correspondence b/w U, V.

Matchings



Input: $G=(U,V,E)$

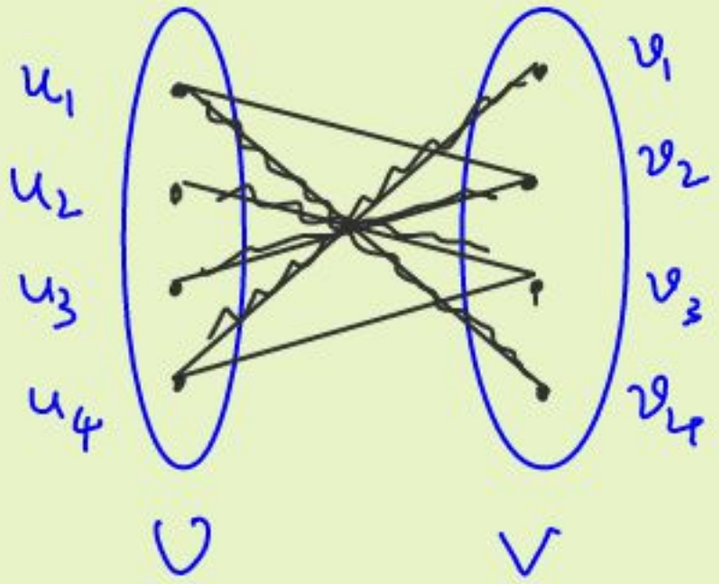
Qn: Does G have a Perfect Matching?

Lovász: Translation to PIT.

$G=(U,V,E)$, $E \subseteq U \times V$

$M \subseteq E$ a Perfect Matching if it defines a 1-1 correspondence b/w U, V.

Matchings



$G = (U, V, E)$, $E \subseteq U \times V$

$M \subseteq E$ a Perfect Matching if it defines a 1-1 correspondence b/w U, V.

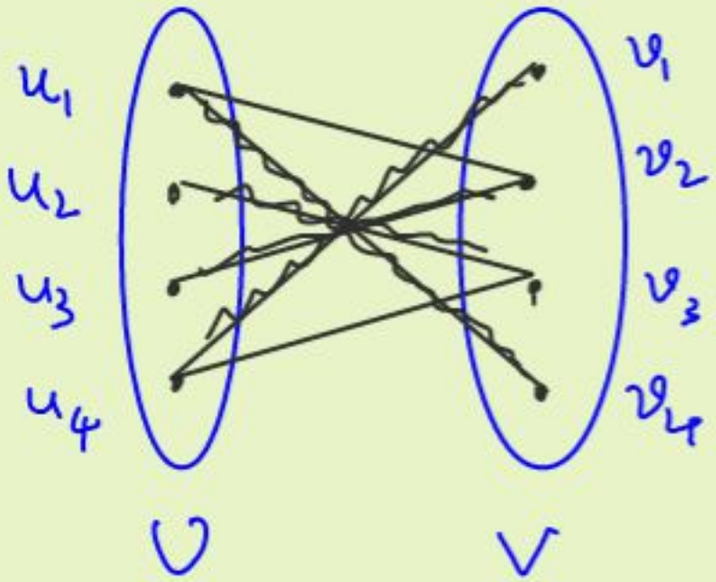
Input: $G = (U, V, E)$

Qn: Does G have a Perfect Matching?

Lovász: Translation to PIT.

$$E_G: \begin{array}{c} \\ u_1 \\ u_2 \\ u_3 \\ u_4 \end{array} \begin{array}{c} \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{pmatrix} 0 & x_{1,2} & 0 & x_{1,4} \\ 0 & 0 & x_{2,3} & 0 \\ 0 & x_{3,2} & 0 & 0 \\ x_{4,1} & 0 & x_{4,3} & 0 \end{pmatrix} \end{array}$$

Matchings



$G = (U, V, E)$, $E \subseteq U \times V$

$M \subseteq E$ a Perfect Matching if it defines a 1-1 correspondence b/w U, V.

Input: $G = (U, V, E)$

Qn: Does G have a Perfect Matching?

Lovász: Translation to PIT.

$$E_G = \begin{array}{c} \\ u_1 \\ u_2 \\ u_3 \\ u_4 \end{array} \begin{array}{c} \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{pmatrix} 0 & x_{1,2} & 0 & x_{1,4} \\ 0 & 0 & x_{2,3} & 0 \\ 0 & x_{3,2} & 0 & 0 \\ x_{4,1} & 0 & x_{4,3} & 0 \end{pmatrix} \end{array}$$

$\det(E_G) \neq 0$ iff G has a P.M.

Deterministic Algorithms for PIT & Lower bounds

→ Rand algo : ① Fix $S \subseteq \mathbb{F}$. $|S| \geq 10d$.

② Pick $a \in_R \mathbb{F}^n$. Test if $P(a) = 0$.

Deterministic Algorithms for PIT & Lower bounds

→ Rand algo : ① Fix $S \subseteq \mathbb{F}$. $|S| \geq 10d$.

② Pick $a \in_R \mathbb{F}^n$. Test if $P(a) = 0$.

→ Can we get efficient non-randomized algo?

Deterministic Algorithms for PIT & Lower bounds

→ Rand algo : ① Fix $S \subseteq \mathbb{F}$. $|S| \geq 10d$.

② Pick $a \in_R \mathbb{F}^n$. Test if $P(a) = 0$.

→ Can we get efficient non-randomized algo?

Thm [Kabanets-Impagliazzo]:

$VP \neq VNP \Rightarrow$ PIT in deterministic subexponential time.

Deterministic Algorithms for PIT & Lower bounds

→ Rand algo: ① Fix $S \subseteq \mathbb{F}$. $|S| \geq 10d$.

② Pick $a \in_R \mathbb{F}^n$. Test if $P(a) = 0$.

→ Can we get efficient non-randomized algo?

Thm [Kabanets-Impagliazzo]:

$VP \neq VNP \Rightarrow$ PIT in deterministic subexponential time.

"Hardness vs. Randomness": Blum-Micali, Yao, Nisan-Wigderson

Much more work since: Dvir-Shpilka-Yehudayoff, Oliveira, Chou-Kumar-Solomon, Guo-Kumar-Saptharishi-Solomon.

State-of-the-art

State-of-the-art

[B-S 83]: Explicit polys $p(x_1, \ldots, x_n)$ requiring circuit size $\Omega(n \log d)$.

[Kalorkoti '85]: Formula lbds of $\Omega(n^3)$.

[CKSV '20]: ABP lbds of $\Omega(n^2)$.

State-of-the-art

[B-S 83]: Explicit polys $p(x_1, \ldots, x_n)$ requiring circuit size $\Omega(n \log d)$.

[Kalorkoti '85]: Formula lbds of $\Omega(n^3)$.

[CKSV '20]: ABP lbds of $\Omega(n^2)$.

More known for restricted models

Homogeneous, Monotone, Multilinear,

Non-commutative, $\Sigma\Pi\Sigma$, $\Sigma\Lambda\Sigma$, ...

Summary

→ Computational models for evaluating polyno-
-mials : Algebraic circuits, formulas, Branching programs.

Summary

→ Computational models for evaluating polyno-
-mials : Algebraic circuits, formulas, Branching programs.

→ Three complexity classes $VF \subseteq VBP \subseteq VP$

Summary

→ Computational models for evaluating polyno-mials: Algebraic circuits, formulas, Branching programs.

→ Three complexity classes $VF \subseteq VBP \subseteq VP$

→ Reductions relate one problem to another. (p-projections)

det_n & $IMM_{n,n}$ VBP-complete.

Summary

→ Computational models for evaluating polyno-mials: Algebraic circuits, formulas, Branching programs.

→ Three complexity classes $VF \subseteq VBP \subseteq VP$

→ Reductions relate one problem to another. (p-projections)

det_n & $IMM_{n,n}$ VBP-complete.

→ VNP - "explicit" polynomials. $VP \subseteq VNP$

per_n - VNP-complete.

Summary

→ Computational models for evaluating polyno-mials: Algebraic circuits, formulas, Branching programs.

→ Three complexity classes $VF \subseteq VBP \subseteq VP$

→ Reductions relate one problem to another. (p-projections)

det_n & $IMM_{n,n}$ VBP-complete.

→ VNP - "explicit" polynomials. $VP \subseteq VNP$

per_n - VNP-complete.

→ Valiant's hypothesis & Perm vs. Det.

Summary

→ Computational models for evaluating polynomials: Algebraic circuits, formulas, Branching programs.

→ Three complexity classes $VF \subseteq VBP \subseteq VP$

→ Reductions relate one problem to another. (p-projections)

det_n & $IMM_{n,n}$ VBP-complete.

→ VNP - "explicit" polynomials. $VP \subseteq VNP$

per_n - VNP-complete.

→ Valiant's hypothesis & Perm vs. Det.

→ Applications to PIT.

Thanks!