A (biased) survey on Arithmetic Circuits

Amir Shpilka
Technion

Goal of talk

Survey results in arithmetic circuit complexity

- Lower Bounds
- Identity Testing
- Reconstruction/Interpolation/Learning

Highlight some `next step' open problems

- Mostly for restricted models
- Show why these models/questions are interesting

Talk outline

- Definition of the Model
- Classical Results
- Lower Bounds
- Identity Testing
- Reconstruction/Interpolation/Learning

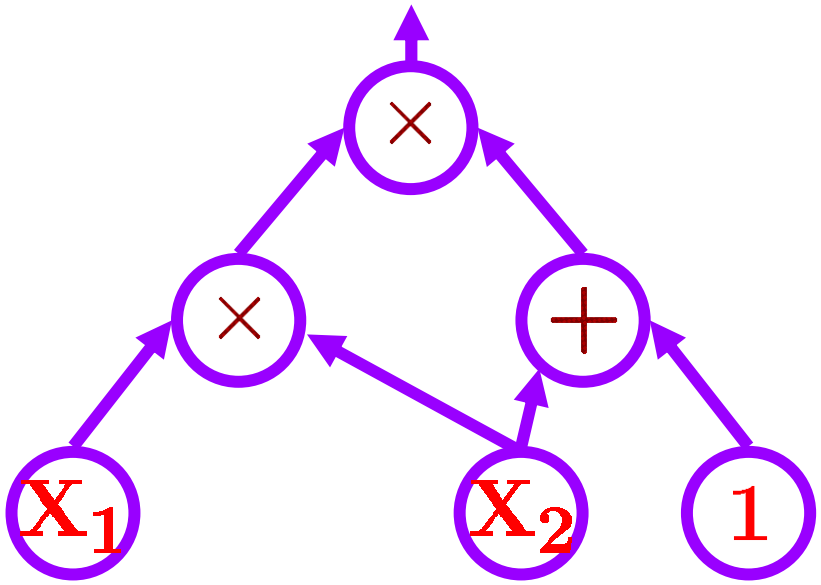
Arithmetic Circuits

Field: $\mathbb{F}$

Variables: $X_1,\ldots,X_n$

Gates: $+$, $\times$

Every gate in the circuit computes a polynomial in $\mathbb{F}[X_1,\ldots,X_n]$



Example: $(X_1 \cdot X_2) \cdot (X_2 + 1)$

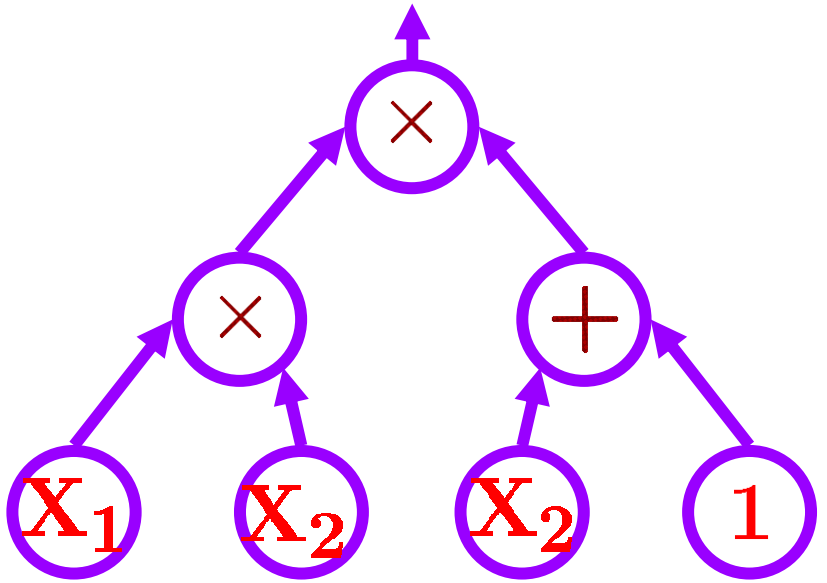
Size = number of gates

Depth = length of longest input-output path

Degree = max degree of internal gates

Arithmetic Formulas

Same, except underlying graph is a tree



Motivation

- Most natural model for computing polynomials
- For many problems (e.g. Matrix Multiplication, Det) best algorithm is by an arithmetic circuit
- Great algorithmic achievements:
  - Fourier Transform
  - Matrix Multiplication
  - Polynomial Factorization
- Structured model (compared to Boolean circuits) $\mathbb{P}$ vs. $\mathbb{NP}$ may be easier
- We like it!

Important Problems

Design new algorithms:

- $\tilde{O}(n^2)$ for Matrix Multiplication?
- Polynomial Factorization without depth increase?

Prove lower bounds:

- Find a polynomial that requires super polynomial size or super logarithmic depth.

Derandomize Polynomial Identity Testing:

- Given (explicitly or as black-boxed) two arithmetic circuits, decide whether they compute the same polynomial.

Reconstruction of arithmetic circuits:

- Compute the circuit from its evaluations.

The classics: Valiant's work

Valiant defined arithmetic analogs of $\mathbb{P}$ and $\mathbb{NP}$:

- $\mathbb{VP}$: All polynomials that have poly size arithmetic circuits of polynomial degree.
- $\mathbb{VNP}$: all polys f that for some g in $\mathbb{VP}$
  $f(x_1,\ldots,x_n)=\Sigma\{g(x_1,\ldots,x_n,e_1,\ldots,e_m) : e_1,\ldots,e_m \in \{0,1\}\}$

Hard problems:

- For all f in $\mathbb{VP}$ there is matrix A, s.t. Det(A) = f, entries of A in $\{\mathbb{F},X_1,\ldots,X_n\}$, size of A is $n^{O(\log n)}$.
- For all f in $\mathbb{VNP}$ $\exists$ matrix A, s.t. Perm(A) = f, entries of A in $\{\mathbb{F},X_1,\ldots,X_n\}$, size of A is poly(n).

The classics: Valiant's work

Valiant defined $\mathbb{VP}$, $\mathbb{VNP}$ and showed that:

- For all f in $\mathbb{VNP}$ there is matrix A, s.t. Perm(A) = f, entries of A in $\{\mathbb{F}, X_1, \ldots, X_n\}$, size of A is poly(n).
- For all f in $\mathbb{VP}$ there is matrix A, s.t. Det(A) = f, entries of A in $\{\mathbb{F}, X_1, \ldots, X_n\}$, size of A is $n^{O(\log n)}$.

To show $\mathbb{VP} \neq \mathbb{VNP}$ enough to prove that any such A with det(A) = Perm(X) has size $n^{\omega(\log n)}$

Best bound [Mignon-Ressayre, Cai-Chen-Li] size(A) = $\Omega(n^2)$

Open Problem 1: Prove size(A) $\geq 2n^2$

The classics cont.

Arithmetic circuits are shallow.

$\mathbb{P}=\mathbb{NC}^2$ [Valiant-Skyum-Berkowitz-Rackoff]: Any size s, deg d circuit can be transformed to a size $poly(s,d)$, deg d, depth $\log(s)\cdot\log(d)$ circuit.

- Low depth version [Agrawal-Vinay]: $size(g) = 2^{o(n)} \Rightarrow g$ has depth-4 circuit of size $2^{o(n)}$
- Proof idea: First apply the V-S-B-R result, then break the circuit to two levels and brute force compute each as a depth-2 circuit.
- Corollary: exponential lower bounds for depth 4 give exponential lower bounds for general circuits

Talk outline

- ✓ Definition of the Model
- ✓ Classical Results
- Lower Bounds
- Identity Testing
- Reconstruction/Interpolation/Learning

Lower Bounds

- Survey known results
- Case of Depth-3 circuits
- Hardness of proving lower bounds?

Lower Bounds

Are actually known!

- Strassen: $\Omega(n \cdot \log d)$ lower bound for computing (simultaneously) $x_1^d, x_2^d, \ldots, x_n^d$
- Baur–Strassen: $\Omega(n \cdot \log d)$ lower bound for computing $y_1x_1^d + y_2x_2^d + \ldots + y_nx_n^d$
- More importantly Baur-Strassen proved: If f has size s, depth d circuit then $\partial f/\partial x_1, \ldots, \partial f/\partial x_n$ have size $O(s)$, depth $O(d)$ circuit.

Open problem 2: What can be said about computing $\{\partial^2 f/\partial x_k \partial x_m\}_{k,m}$?

- If in size $O(s)$, then Matrix Multiplication has $O(n^2)$ algorithm! (consider $\mathbf{x}^t \cdot A \cdot B \cdot \mathbf{y}$)

Multilinear circuits and formulas

Every node computes a multilinear polynomial

[Nisan-Wigderson] exp. Lower bounds for depth 3

[Raz]: PERM requires quasi-poly formulas

mult- $\mathbb{NC}^1 \neq$ mult- $\mathbb{NC}^2$

[Raz-Yehudayoff]: $\exp(n^{\Omega(1/d)})$ bounds for depth d circuits

[Raz-S-Yehudayoff]: $n^{1+\varepsilon}$ lower bound for circuits

Tool: partial derivatives as complexity measure + random restrictions

Open problem 3: exp lower bound for formulas

Open problem 4: quadratic bounds for circuits

Bounded depth circuits

Bounded depth circuits, not a success story…
(recall exponential bounds for $AC^0[p]$ circuits)

- Depth 2 is trivial (sum of monomials)
- Depth 3: exp. lower bounds over char > 0 [Karpinski-Grigoriev, Grigoriev-Razborov]
  Only quadratic lower bounds over $\mathbb{Q}$, $\mathbb{R}$ [S-Wigderson]
- Until recently best lower bounds were nlog**…*n for depth 2d circuits [Pudlak, Raz-S]
- Recently, Raz proved $n^{1+\Omega(1/d)}$ lower bound for depth d circuits.

Situation pathetic, but recall Agrawal-Vinay

Open problem 5: prove a $2n^2$ lower bound for depth-3 circuits over $\mathbb{Q}$, $\mathbb{R}$.

Open problem 6: prove n^3 lower bound for bounded depth circuits over $\mathbb{F}_3$

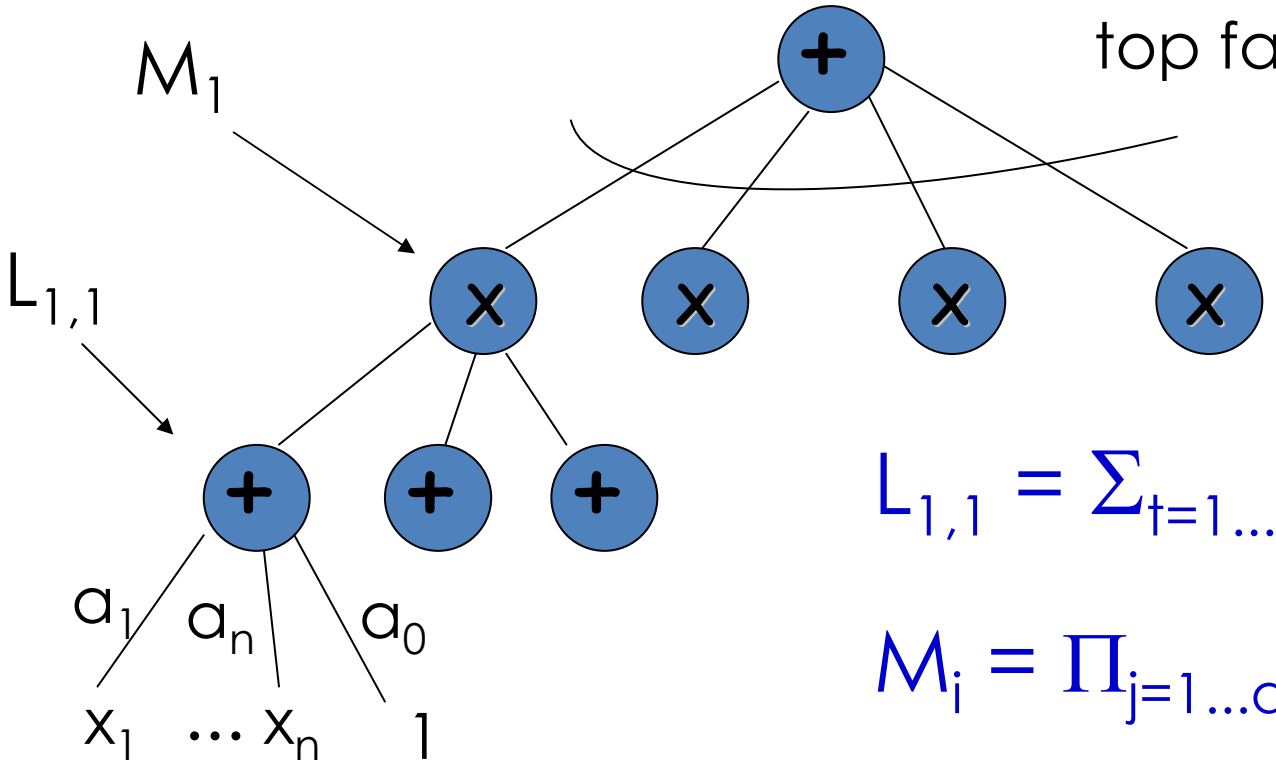
(implies super linear lower bounds for ACC[6])

Lower Bounds

- ✓ Survey known results
- Case of Depth-3 circuits
- Hardness of proving lower bounds?

Depth 3 circuits - $\Sigma\Pi\Sigma(k)$ circuits

Depth-3 = sums of products of linear functions

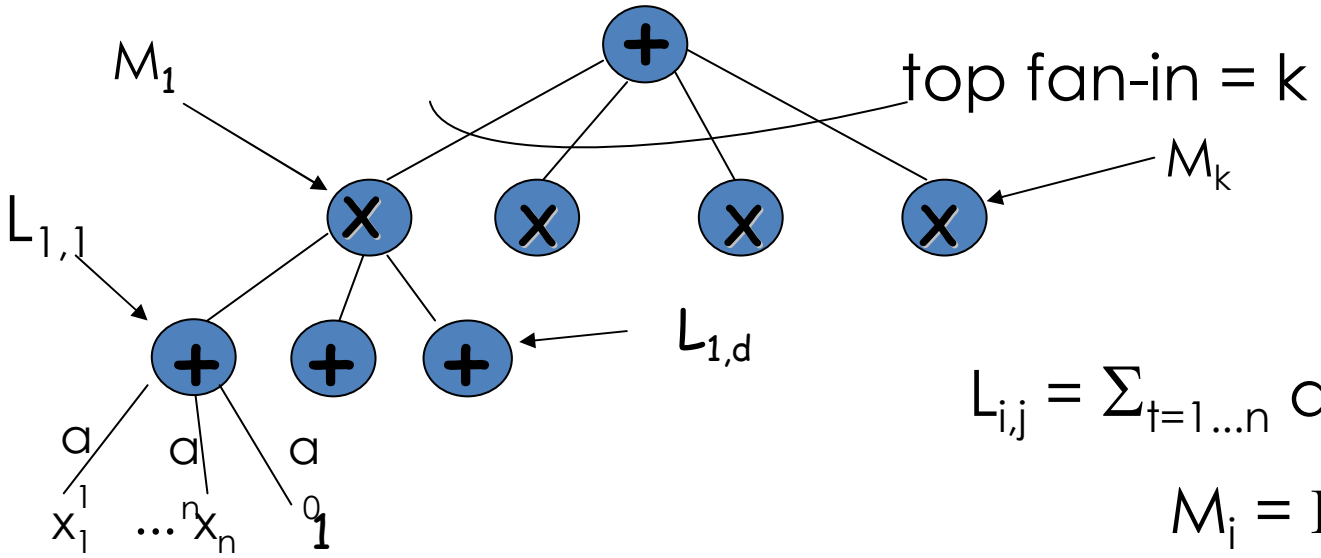


$$L_{1,1} = \Sigma_{t=1\ldots n}\, a_t \cdot x_t + a_0$$

$$M_i = \Pi_{j=1\ldots d_i} L_{i,j}$$

$$C = \Sigma_{i=1\ldots k}\, M_i$$

Depth 3 circuits - ΣΠΣ(k) circuits



$$L_{i,j} = \Sigma_{t=1\ldots n}\, a_t \cdot x_t + a_0$$

$$M_i = \Pi_{j=1\ldots di} L_{i,j}$$

$$C = \Sigma_{i=1\ldots k}\, M_i$$

Alternative view: shifted sparse polynomials

$g = y_1^5\, y_2^3\, y_m^7 + \ldots$ (k monomials)

Replace each variable with a linear function in $\{x_i\}$

Depth-3 Circuits

Grigoriev-Razborov's lower bounds over $\mathbb{F}_p$ (a-la Razborov-Smolensky for $AC^0[p]$ circuits):

- If a multiplication gate contains $n^{1/2}$ linearly independent functions then it is 0, except with probability $\exp(-n^{1/2})$.
- A function in k linear functions has degree $< pk$
- Hence, a circuit with s multiplication gates computes a polynomial that is $s \cdot \exp(-n^{1/2})$ close to a degree $O(n^{1/2})$ polynomial.
- Correlation bounds for Mod(q) give $\exp(-n^{1/2})$ lower bound.

- But what about char 0?

Projections of Symmetric Polynomials

- Let $\sigma_d = \sum_{|T|=d} \prod_{i \in T} x_i$
- Ben-Or: σ_d has $O(n^2)$ depth-3 circuits over $\mathbb{Q}$: Evaluate $f(y)=(y+x_1)\ldots(y+x_n)$ at $n+1$ points, then compute the appropriate linear combination to get the coefficient of y^{n-d} which is σ_d.
- Fact [S]: if $\deg(f) = d$ then there are linear functions $\{L_i\}_{i=1\ldots m}$ such that $f = \sigma_d(L_1,\ldots, L_m)$
- Such f has a size m^2 depth-3 circuit
- Super linear lower bound on m necessary for proving super-quadratic bounds for depth-3
- Best lower bound [S]: $m \geq 2n$ …
- Open problem 7: prove super-linear lower bounds.

Lower Bounds

- ✓ Survey known results
- ✓ Case of Depth-3 circuits
- Hardness of proving lower bounds?

Natural proofs for arithmetic circuits?

- [Razborov-Rudich] A property $\mathcal{P}$ of Boolean functions (truth tables) is natural if:
- Useful against $\mathcal{C}$: If $\mathcal{P}(f) = 1$ then we get a lower bound for circuits from $\mathcal{C}$ computing f.
- Constructivity: There is a $2^{poly(n)}$ sized circuit for computing $\mathcal{P}(f)$ (input is truth table of f).
- Largeness: For "many" functions f, $\mathcal{P}(f) = 1$.
- [R-R]: all known lower bounds are natural.
- [R-R]: PRFGs exist in $\mathcal{C}$ then no strong lower bounds for $\mathcal{C}$(e.g. $\mathcal{C} = TC^0$)

Natural proofs for arithmetic circuits?

Consider multilinear polynomials, given by truth table or list of coefficients.

A property (polynomial) $\mathcal{P}$ is natural if

- Largeness: most f's have $\mathcal{P}$ ($\mathcal{P}(f) \neq 0$)
- Constuctivity: there is a $2^{poly(n)}$ sized arithmetic circuit for computing $\mathcal{P}(f)$.
- Usefulness: gives lower bounds

Note: (most) known proofs are natural

Problem: no PRMPGs are known

(Pseudo-Random-Multilinear-Polynomial-Generator)

[GGM, NR] give polynomials with very high deg

Open Problem 8: Natural proofs for arithmetic circuits

Lower Bounds

- ✓ Survey known results
- ✓ Case of Depth-3 circuits
- ✓ Hardness of proving lower bounds?

Later in this workshop

- Amir Yehudayoff will present structural theorems for arithmetic circuits and formulas with application to lower bounds.

- Ran Raz will present several approaches to proving lower bounds including Elusive Polynomial Mappings.

Did not cover

- Linear complexity (FFT, Gaussian elimination)
- Valiant's rigidity approach
- Bilinear complexity (matrix multiplication)
- Kalorkoti's Lower bounds for arithmetic formulas
- Monotone circuits

Outline of Talk

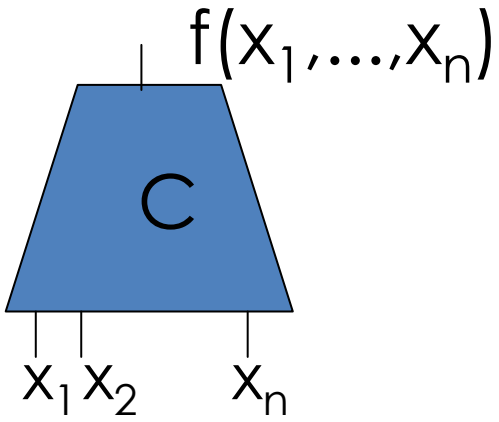
- ✓ Definition of the Model + some context
- ✓ Classical Results
- ✓ Lower Bounds
- Identity Testing
- Reconstruction/Interpolation/Learning

Polynomial Identity Testing

- Definition of the problem
- Connection to lower bounds (hardness)
  - Kabanets-Implagliazzo
  - Agrawal
  - Dvir-S-Yehudayoff
- Survey of positive results
- Some proofs:
  - Sparse polynomials
  - Partial derivatives technique
  - Depth-3 circuits
  - Read-Once formulas
- Connection to polynomial factorization

Polynomial Identity Testing

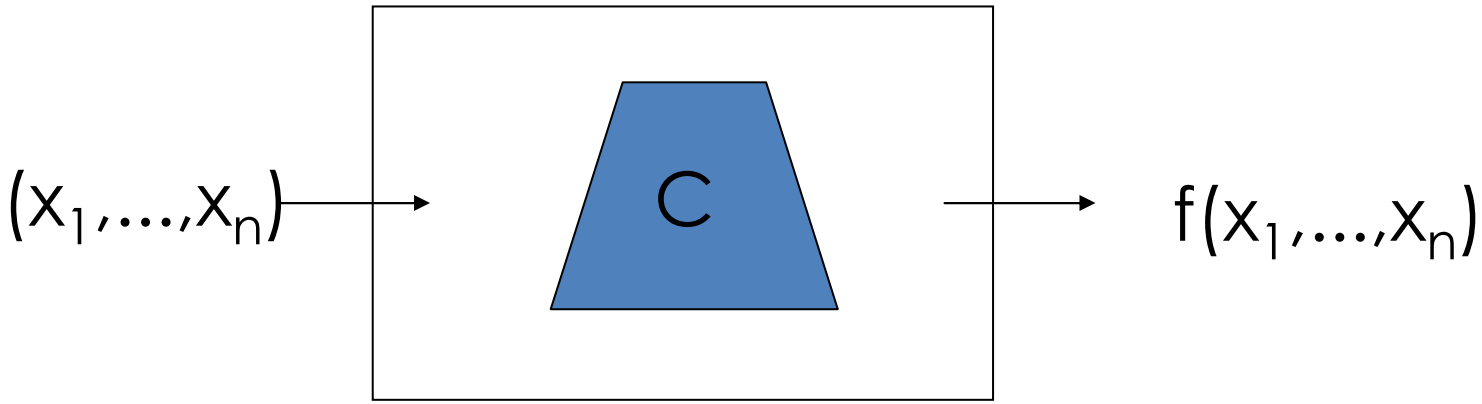
Input: Arithmetic circuit computing f
Problem: Does f=0 ?



Randomized algorithm [Schwartz, Zippel, DeMillo-Lipton]: evaluate f at a random point (more on that later)
Goal: deterministic algorithm

Black Box PIT ≡ Hitting Sets

Input: A Black-Box circuit computing f.
Problem: Does f=0 ?



Goal: deterministic algorithm (a.k.a. Hitting Set)
S,Z,DM-L: ∃ small Hitting Set (not explicit)

Motivation

- Natural and fundamental problem
- In some sense: most general problem in co-$\mathbb{RP}$
- Strong connection to circuit lower bounds
- Algorithmic importance:
  - Primality testing [Agrawal-Kayal-Saxena]
  - Parallel algorithms for finding matching [Karp-Upfal-Wigderson, Mulmuley-Vazirani-Vazirani]

Polynomial Identity Testing

- ✓ Definition of the problem
- Connection to lower bounds (hardness)
  - Agrawal
  - Kabanets-Implagliazzo
  - Dvir-S-Yehudayoff
- Survey of positive results
- Some proofs:
  - Sparse polynomials
  - Partial derivatives technique
  - Depth-3 circuits
  - Read-Once formulas
- Connection to polynomial factorization

Hardness: PIT ≡ lower bounds

[Kabanets-Impagliazzo]:

- $2^{\Omega(n)}$ lower bound for Permanent $\Rightarrow$ PIT in $n^{\text{polylog } n}$ time
- PIT $\in$ P $\Rightarrow$ super-polynomial lower bounds: Boolean for $\mathbb{NEXP}$, or arithmetic for Permanent

[Dvir-S-Yehudayoff]: (almost) same as K-I for bounded depth circuits

[Agrawal]: Black-Box PIT $\Rightarrow$ lower bounds for arithmetic circuits

Lesson: derandomizing PIT essentially equivalent to proving arithmetic circuit lower bounds

Black-Box P.I.T. $\Rightarrow$ Lower Bounds

[Agrawal]: Black-Box P.I.T for size s circuits in time $poly(s)$ (i.e. $poly(n)$ size hitting set) implies exponential lower bounds for arithmetic circuits:

Given $H=\{p_i\}$, find non-zero $O(\log(s))$-variate polynomial f such that $f(p_i)=0$ for all i.

$\Rightarrow$ f does not have size s circuits

Gives lower bounds for f in **PSPACE**

Conjecture [Agrawal]:

$H=\{(y_1,\ldots, y_n) : y_i=y^{ki \bmod r}, k,r < s^{20}\}$ is a hitting set for size s circuits

Non Black-Box P.I.T. $\Leftrightarrow$ Lower Bounds [K-I]

[Valiant,Toda,Impagliazzo-Kabanets-Wigderson]:

$\mathbb{NEXP} \subseteq \mathbb{P}$/Poly $\Rightarrow$ Perm is $\mathbb{NEXP}$-complete

$\Rightarrow$: Perm has poly size arith. circuit $\Rightarrow$ Perm in $\mathbb{NP}^{\text{PIT}}$

Idea: guess circuit for Perm. verify correctness using self reducibility and PIT.

$\Rightarrow$: If $\mathbb{NEXP} \subseteq \mathbb{P}$/Poly and Perm has poly size circuits and PIT in $\mathbb{P}$ then $\mathbb{NEXP}$ in $\mathbb{NP}$ $\Rightarrow\Leftarrow$

Other direction follows by using arithmetic version of N-W generator and Kaltofen's factoring results.

Non Black-Box P.I.T. $\Leftrightarrow$ Lower Bounds in bounded depth circuits

[Dvir-S-Yehudayoff]: $2^{n^{\varepsilon}}$ lower bound for depth d $\Rightarrow$ PIT of depth d-5 in $n^{\text{polylog } n}$ time

Caveat: requires the polynomial computed by the circuit to have small individual degrees.

Inverse direction (a-la K-I) as before.

Main tool: assume $P(x_1,\ldots,x_n,y)$ has size s depth d circuit. Let $f(x_1,\ldots,x_n)$ be a root of P i.e. $P(x_1,\ldots,x_n,f(x_1,\ldots,x_n)) \equiv 0$.

Then f has circuit of size $\text{poly}(s + m^r)$ and depth d+3, where $m = \deg(f)$ and $r = \deg_y(P)$.

Open problem 9: remove degree restriction.

Polynomial Identity Testing

- ✓ Definition of the problem
- ✓ Connection to lower bounds (hardness)
  - ✓ Agrawal
  - ✓ Kabanets-Implagliazzo
  - ✓ Dvir-S-Yehudayoff
- Survey of positive results
- Some proofs:
  - Sparse polynomials
  - Partial derivatives technique
  - Depth-3 circuits
  - Read-Once formulas
- Connection to polynomial factorization

Randomized algorithms for PIT

Schwartz-Zippel-DeMillo-Lipton:

Evaluate C at a random input

Gives error-randomness tradeoff

Chen-Kao: trade time for error over $\mathbb{R}$:

$\pi_i = \pm p_{i,1}^{\frac{1}{2}} \ldots \pm p_{i,r}^{\frac{1}{2}}$, for different primes, random signs

Then $C \equiv 0$ iff $C(\pi_1, \pi_2, \ldots, \pi_n) = 0$

Truncating after t digits gives error $O(1/t)$

Intuition: random conjugate won't vanish mod 2^{-t}

For multilinear polynomials, C-K use n random bits for $1/\text{poly}$ error, S-Z-DM-L use $n\log(n)$ bits for error $\frac{1}{2}$.

Lewin-Vadhan: generalized to finite fields:
irreducible polynomials $\leftrightarrow$ primes,
power series $\leftrightarrow$ square roots. Truncation mod x^t.

Randomized algorithms for PIT

Agrawal-Biswas:

Observe: $C \equiv 0$ iff $C(y, y^D, y^{D^2}, \ldots, y^{D^n}) \equiv 0$

Problem: degree too large

A-B give a "small" set of polynomials $\{f_i(y)\}$ s.t.
$C \equiv 0$ iff $\forall i\ C(y, y^D, y^{D^2}, \ldots, y^{D^n}) \equiv 0 \bmod f_i(y)$

Similar idea used in primality test of
Agrawal-Kayal-Saxena

Uses less random bits than S-Z-DM-L

Agrawal's conjecture:

$\{(y_1, \ldots, y_n) : y_i = y^{k_i \bmod r}, k, r < s^{20}\}$ is a hitting set for size s circuits

Deterministic algorithms for PIT

- Depth-2 circuits (sparse polys) [BenOr-Tiwari, Grigoriev-Karpinski, Klivans-Spielman,…]
  - Black-Box in polynomial time
- Non-commutative formulas [Raz-S]
  - Non-Black-Box in polynomial time
- $\Sigma\Pi\Sigma(k)$ circuits [Dvir-S, Kayal-Saxena, Arvind-Mukhopadhyay, Karnin-S, Saxena-Seshadri, Kayal-Saraf]
  - Black-Box in quasi-polynomial time*
  - Non-Black-Box in time $n^{O(k)}$
- Sum of k Read-once formulas [S-Volkovich]
  - Black-Box in $n^{O(\log(n)+k)}$
  - Non-Black-Box in time $n^{O(k)}$.

Why study restricted models

- [Agrawal-Vinay] PIT for depth-4 implies PIT for general depth.
- Gaining insight to more general questions:

  Intuitively: lower bounds imply PIT

  Multilinear formulas: super polynomial bounds [Raz] but no PIT algorithm

  Not even for Depth-3 multilinear formulas!

  Sum of ROFs, depth-3 multilinear formulas relaxations of the more general problem
- Interesting results: Structural theorem for $\Sigma\Pi\Sigma(k)$
- Better ideas needed! That's why you're here!

Polynomial Identity Testing

- ✓ Definition of the problem
- ✓ Connection to lower bounds (hardness)
  - ✓ Kabanets-Implagliazzo
  - ✓ Agrawal
  - ✓ Dvir-S-Yehudayoff
- ✓ Survey of positive results
- Some proofs:
  - Sparse polynomials
  - Partial derivatives technique
  - Depth-3 circuits
  - Read-Once formulas
- Connection to polynomial factorization

Depth-2 circuits [Klivans-Speilman]

$f(x_1,\ldots,x_n) = M_1 + \ldots + M_m$ sum of m monomials of degree d.

Idea: replace x_i by y^{a_i} such that all monomials map uniquely, interpolate resulting polynomial.

Problem: a_i–s need to grow fast (gives high degree)

Solution: for p prime $> d^2/\varepsilon$ and $k \le p$ set $a_i = k^{i-1} \bmod p$. Evaluate at $np+1$ different y-s

$x_1^{e_1} \cdot x_2^{e_2} \cdot \ldots \cdot x_n^{e_n}$ mapped to $y\wedge(e_1+e_2k+\ldots+e_nk^{n-1}) = y^{E(k)}$

m monomials define m polynomials $E_1(k), \ldots, E_m(k)$. They are mapped 1-1 if k is not root of any E_i-E_j. Holds for all but ε fraction of the k's.

Better constructions are known

Non commutative formulas

Special case: set-multilinear depth-3 circuits
$X = X_1 \sqcup X_2 \sqcup \ldots \sqcup X_d$, $X_i = \{x_{i,1}, \ldots, x_{i,n}\}$

Multiplication gate: $M_i = L_{i,1}(X_1) \cdot \ldots \cdot L_{i,d}(X_d)$

$C = M_1 + \ldots + M_s$

Main observation: dimension of partial derivatives of C according to $X_1, \ldots, X_k$ (any k) is at most s (spanned by $L_{i,k+1}(X_{k+1}) \cdot \ldots \cdot L_{i,d}(X_d)$)

Algorithm: compute a basis for all derivatives according to $X_1, \ldots, X_k$ starting from $k=1$ to $k=d$. $C \equiv 0$ if at the end all basis elements are 0

Same idea also in the general case

Depth-3 circuits ($\Sigma\Pi\Sigma(k)$ circuits)

$L = \Sigma_{t=1\ldots n}\, a_t \cdot x_t + a_0$, $M_i = \Pi_{j=1\ldots d_i} L_{i,j}$, $C = \Sigma_{i=1\ldots k}\, M_i$

Definition:

C simple if no linear function appears in all the M_i-s

C minimal if no subset of mult. gates sums to zero

Main tool [Dvir S]: If $C \equiv 0$ simple and minimal then $\dim(\text{span}(L_{i,j})) \leq R(k,d) = (\log(d))^{k*}$

Lesson: If $C \equiv 0$ then it is very structured

Non Black-Box Algorithm: find partition to sub-circuits of low dimension (after removal of g.c.d.) and brute force verify that they vanish.

Improved $n^{O(k)}$ algorithm by [Kayal-Saxena].

Black-Box PIT for $\Sigma\Pi\Sigma(k)$

Black-Box algorithm [Karnin-S]: restrict C to a low dim subspace such that dimension of sub-circuits does not fall too much.

Claim: such map preserve structure of C

Claim: $C|_V \equiv 0$ iff $C \equiv 0$

Can find poly set of V-s of dimension $O(R(k,d))$

Gives: $poly(n) \cdot d^{O(R(k,d))}$ time algorithm

[Saxena-Seshadri]: finite $\mathbb{F}$, $R(k,d) < k^3 \log(d)$

[Kayal-Saraf]: over $\mathbb{R}$, $R(k,d) < \exp(k)$

Improve [Dvir-S] and [Karnin-S] (plug and play)

Proofs: not today…

Read-Once formulas (ROFs)

A formula where every variable labels at most one leaf.

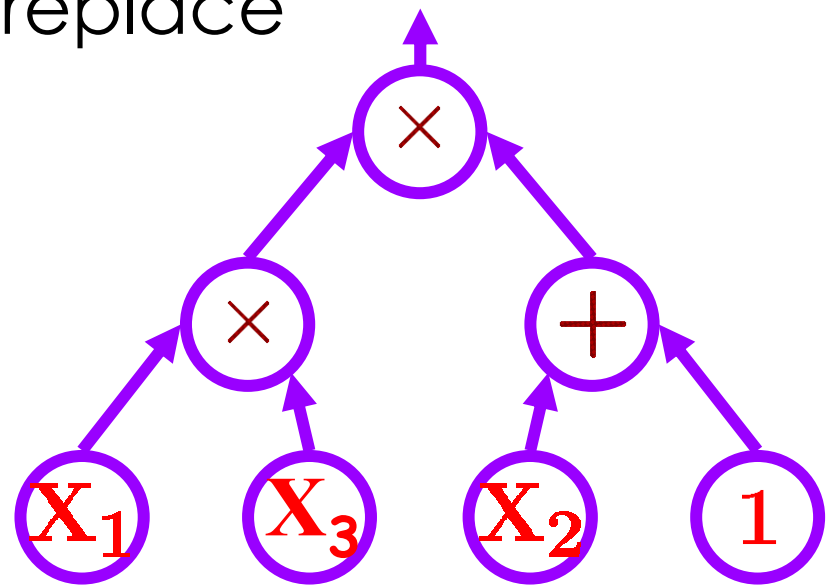
Preprocessed ROF: can replace each x_i with $T_i(x_i)$

Sum of ROFs:

$F = F_1 + F_2 + \ldots + F_k$

each F_i is a (P-)ROF

Result: Black-Box PIT for sum of k (P)ROFs in time $n^{O(\log(n)+k)}$



Generator for ROFs

$A = \{\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_n\} \subseteq \mathbb{F}$

$u_i(x)$ be such that $u_i(a_j) = \delta(i,j)$

Def: For every $i \in [n]$ and $k \geq 1$:

$G^i_k(\mathbf{y},\mathbf{z}) \triangleq u_i(y_1)\cdot z_1 + u_i(y_2)\cdot z_2 + \ldots + u_i(y_k)\cdot z_k$

$G_k(\mathbf{y},\mathbf{z}) \triangleq (G^1_k, G^2_k, \ldots, G^n_k)$

Crucial Property: $G_k \mid_{(y_k = \alpha_m)} = G_{k-1} + z_k \cdot \bar{e}_m$

Enables us to isolate any single coordinate

PIT for PROFs

Theorem: Let P be a non-zero PROP then $P(G_{\log(n)+1}) \neq 0$. Moreover, if P is a non-constant polynomial then so is $P(G_{\log(n)+1})$

Proof idea: induction on structure of formula.

If the top gate is $\times$ then by induction we are ok. If top gate is +, then one son has few variables.

Can keep a variable that belongs to small son 'alive'.

Sum of ROFs

$F = F_1 + F_2 + \ldots + F_k$

Idea: PIT for ROFs gives a justifying set for any k ROFs of size $n^{O(\log n)}$

Justifying set: contains at least one input $(a_1,\ldots,a_n)$ such that if F_i depends on x_m then $F_i(a_1,\ldots,a_{m-1},x_m,a_{m+1},\ldots,a_n)$ depends on x_m .

By changing $x_i \leftarrow x_i+a_i$ assume that all the F_i-s are **0**-justifyied.

I.e. assigning zeros to all variables but x_i keeps dependence on x_i

Hardness of representation

Hardness of representation: no sum of $k < n/3$ 0-justified ROFs can compute $x_1 \cdot x_2 \cdot \ldots \cdot x_n$

Proof Idea: By induction on k. By taking partial derivatives and making substitutions, can remove some of the ROFs but preserve the structures of F and $x_1 \cdot x_2 \cdot \ldots \cdot x_n$.

Theorem: Let F be a sum of k 0-justified ROFs. Let $\mathcal{A}$ be a set of all vectors in $\{0,1\}^n$ of Hamming weight $\leq k$. Then $F \equiv 0 \Leftrightarrow F|_{\mathcal{A}} \equiv 0$.

Idea: For $n \leq k$ clear. For large n, set $x_i = 0$. Induction implies $x_i \mid F$. Hence $x_1 \cdot x_2 \cdot \ldots \cdot x_n \mid F$

$\Rightarrow\Leftarrow$

Some open problems

- Open problem 10: Give a Black-Box PIT algorithm for non-commutative formulas
- Open problem 11: Solve depth-3 case already!
- Open problem 12: Solve multilinear depth-3 case already!
- Open problem 13: Polynomial time PIT for (sum of) (P-)ROFs
- Open problem 14: $f(k)\cdot n^{O(1)}$ time PIT for sum of k ROFs

Polynomial Identity Testing

- ✓ Definition of the problem
- ✓ Connection to lower bounds (hardness)
  - ✓ Kabanets-Implagliazzo
  - ✓ Agrawal
  - ✓ Dvir-S-Yehudayoff
- ✓ Survey of positive results
- ✓ Some proofs:
  - ✓ Sparse polynomials
  - ✓ Partial derivatives technique
  - ✓ Depth-3 circuits
  - ✓ Read-Once formulas
- Connection to polynomial factorization

PIT and Factoring

f is *composed* if $f(X) = g(X|_S)\cdot h(X|_T)$ where S and T are disjoint

[S-Volkovich]: PIT is equivalent to factoring to *decomposable* factors.

$\Leftarrow$: $f \equiv 0$ iff $f+y\cdot z$ has two decomposable factors.

$\Rightarrow$: Claim: If we have a (BB or NBB) PIT for all circuits of the form $C_1 + C_2\cdot C_3$, where $C_i \in \mathcal{M}$ then given (BB or NBB) $C \in \mathcal{M}$ we can deterministically output (BB or NBB) all decomposable factors of C.

Decomposable factoring using PIT

Claim: If we have a (BB or NBB) PIT for all circuits of the form $C_1 + C_2 \cdot C_3$, where $C_i \in \mathcal{M}$ then given (BB or NBB) $C \in \mathcal{M}$ we can deterministically output (BB or NBB) all decomposable factors of C.

Idea: Using PIT find a justifying assignment $\mathbf{a}$ for C. Set $x_n = a_n$ and factor (recursively).

Assume $S_1, \ldots, S_k$ is the partition of $[n-1]$.

For every S_i check weather

$$C(\mathbf{a}) \cdot C \equiv C(X_{Si} \leftarrow a_{Si}) \cdot C(X_{[n]\setminus Si} \leftarrow a_{[n]\setminus Si})$$

If yes, add S_i to the partition. At the end put all the remaining vars in a new set.

PIT and factoring

- Deterministic decomposable factoring is equivalent to lower bounds:
  - Deterministic factoring implies $\mathbb{NEXP}$ does not have small arithmetic circuits
  - Lower bounds imply Deterministic decomposable factoring
- PIT $\equiv$ factoring for multilinear polynomials
- Deterministic decomposable factoring for depth-2, $\Sigma\Pi\Sigma(k)$, sum of read-once…
- New: subexponential PIT for multilinear $\Sigma\Pi\Sigma\Pi(k)$ circuits (depth-4)
- Open problem 15: is PIT equivalent to general factorization?

Polynomial Identity Testing

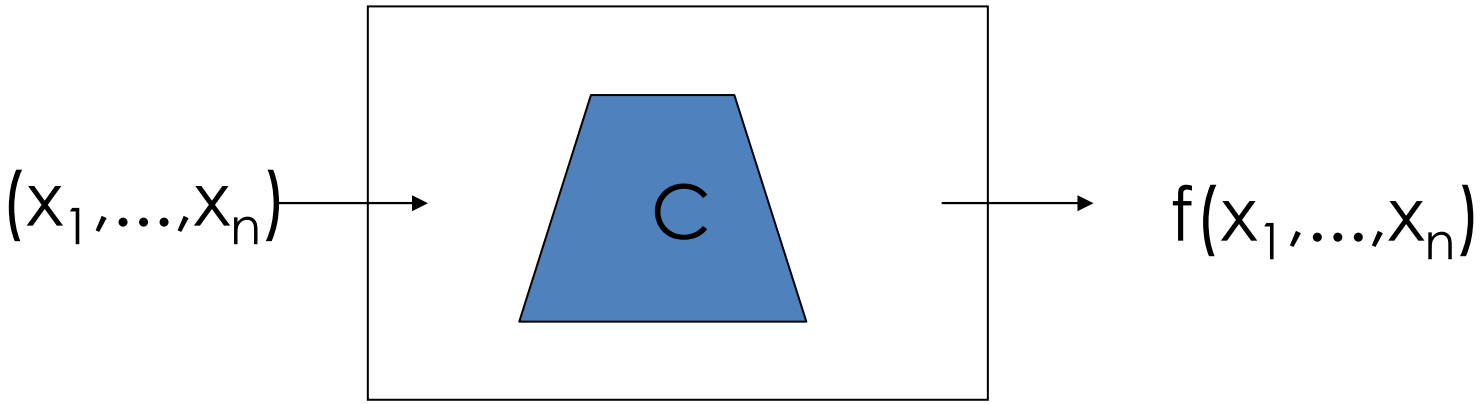
- ✓ Definition of the problem
- ✓ Connection to lower bounds (hardness)
  - ✓ Kabanets-Implagliazzo
  - ✓ Agrawal
  - ✓ Dvir-S-Yehudayoff
- ✓ Survey of positive results
- ✓ Some proofs:
  - ✓ Sparse polynomials
  - ✓ Partial derivatives technique
  - ✓ Depth-3 circuits
  - ✓ Read-Once formulas
- ✓ Connection to polynomial factorization

Reconstructing Arithmetic Circuits

- Definition of the problem
- Connection to Learning of Boolean functions
- Survey of known results
- Some proofs?

Reconstruction of arithmetic circuits

Input: Black-Box arithmetic circuit, over a finite field $\mathbb{F}$, computing a polynomial f



Goal: Find a small circuit for f, using few queries (from $\mathbb{F}^n$ or extension field of $\mathbb{F}$)

Motivation: natural problem,
algebraic analog of learning

Reconstruction vs. Learning

Learning: queries/examples over {0,1}

Reconstruction: queries over □, a much richer domain (e.g. interpolation is possible)

Learning: arbitrary distributions

Reconstruction: uniform distribution

Learning with queries: under crypto queries can't help.

Reconstruction: no such results known, same difficulty as in generalizing natural proofs.

Learning with queries: usually want to output a close enough hypothesis.

Reconstruction: two close polynomials are equal.

Exact learning: hardness of proper learning known

Reconstruction: still no **strong** hardness-results.

Positive results

- Depth-2 circuits: can reconstruct in polynomial time over any field [BenOr-Tiwari, Grigoriev-Karpinski, Klivans-Spielman,…].
- Classes computing polynomials with "few" partial derivatives (e.g. set-multilinear $\Sigma\Pi\Sigma$): poly time using multiplicity-automata [Beimel-Bergadano-Bshouty-Kushilevitz-Varricchio,Klivans-S].
- $\Sigma\Pi\Sigma(k)$ circuits: quasi-polynomial [S,Karnin-S].
- Read-Once formulas:
  - Randomized poly time: [Hancok-Hellerstein,Hancok-Hellerstein-Bshouty,Bshouty-Bshouty].
  - Deterministic quasi-polynomial time [S-Volkovich].

Hardness results

- [Fortnow-Klivans] a-la Kabanets-Impagliazzo: If there is a PAC+MQ learning algorithm for C then there is an exponential lower bound for a polynomial computed in $ZPEXP^{RP}$
- Connects lower bounds to learning arithmetic circuits (even over uniform distribution).
- [Klivans-Sherstov]: Under crypto assumptions no PAC learning algorithm for depth-3 circuits.
- Downside: algorithm should work for any distribution (not just uniform), can't use queries.
- Open problem 16: prove better hardness results for learning arithmetic circuits under uniform dist.

Sparse reconstruction

- Idea: play with substitutions to x_i to learn its degree in each monomials
- Recall: $a_i = k^{i-1} \bmod p$, for p prime $> d^2/\varepsilon$ and $k=1,\ldots,p$. Substitute $x_i \leftarrow y^{a_i}$. Evaluate at $np+1$ different y-s.
- For some i set, $a_i = 2y^{a_i}$
- Changes the coefficient of each relevant monomial.

$\Sigma\Pi\Sigma(2)$ reconstruction

- If rank is low then brute force learn
- If rank is high then project to a random subspace (keeps rank high)
- Brute force find many (O(log n)) linearly independent functions that have higher multiplicity in M_1 than in M_2.
- Work "modulo" those functions and reconstruct M_2
- Now reconstruct M_1
- Lift circuit to the whole space

Summary of Talk

- ✓ Definition of the Model + some context
- ✓ "Classical Results"
- ✓ Lower Bounds
- ✓ Identity Testing
- ✓ Reconstruction/Interpolation/Learning
- Some more things that we did not cover
- Open problems

More things we did not cover

- Algorithms:
  - Matrix Multiplication (wait for Chris Umans' talk!)
  - Determinant in NC^2
  - Polynomial Factorization
- Complexity of univariate polynomials:
  - How many operations are needed to compute $(x+1)\cdot(x+2)\cdot\ldots\cdot(x+n)$?
- Other restricted models:
  - Bounded coefficients circuits
  - Monotone circuits

Summary of open problems

1. Show: if Det(A) = Perm(X) then size(A) = $n^{\omega(1)}$
2. What can be said about computing $\{\partial f/\partial x_k \partial x_m\}_{k,m}$? (generalizing Baur-Strassen)

$2\frac{1}{2}$. What about computing $\{\partial^2 f/\partial x_k \partial x_k\}_k$?

3. Prove super-quadratic lower bounds for depth-3 circuits over □, □.
4. Prove n^3 lower bounds for bounded depth circuits over $\mathbb{F}_3$.
5. Prove lower bounds for $\sigma_d(L_1,\ldots, L_m)$
6. Lower bounds for non-commutative circuits
7. Exponential lower bound for multilinear formulas
8. Separation of multilinear and non-multilinear formula size

Summary of open problems

9. Super-poly lower bound for multilinear circuits
10. Natural proofs for arithmetic circuits.
11. Factor without increasing depth of circuit
12. Remove degree restriction from root finding algorithm
13. Give a Black-Box PIT algorithm for non-commutative formulas
14. Solve PIT for depth-3 circuits
15. Solve PIT for multilinear depth-3 circuits
16. Polynomial time PIT for (sum of) ROFs
17. Is PIT equivalent to general factorization?

More open questions

18. Prove (more) hardness of learning arithmetic circuits
19. Lower bounds for depth-3 circuits where each linear function involves just 5 vars. (when 5=2 we can do it)
20. P.I.T. for same model
21. P.I.T. for the symmetric model
22. P.I.T. for depth-4 with restricted fan-in (can do for multilinear circuits)
23. P.I.T. for read-k formulas (can do it for k=2)
24. Reconstruction of sum of ROF's
25. …

Thank You!