On Approximate Polynomial Identity Testing and Real Root Finding

Gorav Jindal

Saarland Informatics Campus, Saarbrücken

November 11, 2019

Outline

Outline

Based on

- Joint work with Prof. Dr. Markus Bläser and Anurag Pandey.
- Publications:
  - *Greedy Strikes Again: A Deterministic PTAS for Commutative Rank of Matrix Spaces* Bläser, Markus, Jindal, Gorav, and Pandey, Anurag **In 32nd Computational Complexity Conference** (CCC 2017).
  - *A Deterministic PTAS for the Commutative Rank of Matrix Spaces* Bläser, Markus, Jindal, Gorav, and Pandey, Anurag **In Theory of Computing** 2018.

Matrix Spaces

Definition (Matrix Space)

A vector space $\mathcal{B} \leq \mathbb{F}^{n \times n}$ is called a *matrix space:*

$$\mathcal{B} = \langle B_1, B_2, \ldots, B_m \rangle.$$

- Here $B_1, B_2, \ldots, B_m$ linearly generate $\mathcal{B}$.

Matrix Spaces

Definition (Matrix Space)

A vector space $\mathcal{B} \leq \mathbb{F}^{n \times n}$ is called a *matrix space:*

$$\mathcal{B} = \langle B_1, B_2, \ldots, B_m \rangle.$$

- Here $B_1, B_2, \ldots, B_m$ linearly generate $\mathcal{B}$.

Definition (Commutative rank)

For a matrix space $\mathcal{B}$, maximum rank of any matrix in $\mathcal{B}$ is the *commutative rank* of $\mathcal{B}$, use $\text{crk}(\mathcal{B})$ to denote it.

Symbolic Matrices

Definition (Symbolic Matrix)

A matrix $B \in (\mathbb{F}[x_1, x_2, \ldots, x_m])^{n \times n}$ whose entries are homogeneous linear forms is called a *symbolic matrix.*

Symbolic Matrices

Definition (Symbolic Matrix)

A matrix $B \in (\mathbb{F}[x_1, x_2, \ldots, x_m])^{n \times n}$ whose entries are homogeneous linear forms is called a *symbolic matrix.*

- Use $\text{rank}(B)$ to denote the rank of B over $\mathbb{F}(x_1, x_2, \ldots, x_m)$.
- Matrix space $\mathcal{B} = \langle B_1, B_2, \ldots, B_m \rangle$, associate a symbolic matrix B with $\mathcal{B}$ by:

$$B \overset{\text{def}}{=\!=} \sum_{i=1}^{m} x_i B_i .$$

Rank Connection of Symbolic Matrices and Matrix Spaces

Theorem (Folklore)

$\mathcal{B} = \langle B_1, B_2, \ldots, B_m \rangle \leq \mathbb{F}^{n \times n}$ *a matrix space and*

$$B(x_1, x_2, \ldots, x_m) \stackrel{def}{=\!=} \sum_{i=1}^{m} x_i B_i$$

the corresponding symbolic matrix, then

$$\mathrm{rank}(B) = \mathrm{crk}(\mathcal{B}).$$

(Assuming $|\mathbb{F}| > n$).

Maximum Matching to Commutative rank

- Tutte matrix A_G of a simple undirected graph $G = (V, E)$ with $V = [n]$ is an $n \times n$ symbolic matrix defined as:
- $$(A_G)_{i,j} = \begin{cases} x_{ij} & \text{If } (i,j) \in E \text{ and } i < j \\ -x_{ji} & \text{If } (i,j) \in E \text{ and } i > j \\ 0 & \text{Otherwise} \end{cases}$$

Maximum Matching to Commutative rank

- Tutte matrix A_G of a simple undirected graph $G = (V, E)$ with $V = [n]$ is an $n \times n$ symbolic matrix defined as:
- $$(A_G)_{i,j} = \begin{cases} x_{ij} & \text{If } (i,j) \in E \text{ and } i < j \\ -x_{ji} & \text{If } (i,j) \in E \text{ and } i > j \\ 0 & \text{Otherwise} \end{cases}$$

Theorem (Lovász 1979)

If r is the size of maximum matching in G then $\text{rank}(A_G) = 2r$.

Polynomial Identity Testing (PIT) Using Commutative rank

Problem

(FORMULA PIT) A formula F computing $f \in \mathbb{F}[x_1, x_2, \ldots, x_m]$, is $f = 0$?

Polynomial Identity Testing (PIT) Using Commutative rank

Problem

(FORMULA PIT) A formula F computing $f \in \mathbb{F}[x_1, x_2, \ldots, x_m]$, is $f = 0$?

Theorem (Valiant 1979)

If $f \in \mathbb{F}[x_1, x_2, \ldots, x_m]$ is computed by a formula of size s then one can compute (in deterministic $\mathrm{poly}(m, s)$ time) an affine symbolic matrix F of size $(s+2) \times (s+2)$ such that $\det(F) = f$.

- Checking the non-zeroness of f reduces to checking if the symbolic matrix F has full rank.

Outline

Computing the Commutative Rank

- To compute the commutative rank exactly, an easy randomized algorithm exists.
  - Substitute random field scalars for x_i's and compute the rank of the resulting scalar matrix.

Computing the Commutative Rank

- To compute the commutative rank exactly, an easy randomized algorithm exists.
  - Substitute random field scalars for x_i's and compute the rank of the resulting scalar matrix.
- Deterministically computing the commutative rank leads to deterministic PIT.

Computing the Commutative Rank

- To compute the commutative rank exactly, an easy randomized algorithm exists.
  - Substitute random field scalars for x_i's and compute the rank of the resulting scalar matrix.
- Deterministically computing the commutative rank leads to deterministic PIT.
- Approximating the commutative rank deterministically?

Approximating the Commutative Rank

- A related notion of the non-commutative rank $\mathrm{ncrk}(\mathcal{B})$ of a matrix space $\mathcal{B} \leq \mathbb{F}^{n \times n}$.

Theorem (Fortin, Reutenauer 2004)

If $\mathbb{F}$ is an infinite field then:

$$\mathrm{crk}(\mathcal{B}) \leq \mathrm{ncrk}(\mathcal{B}) \leq 2 \cdot \mathrm{crk}(\mathcal{B}).$$

- Above inequalities are tight.

Approximating the Commutative Rank

Theorem (GGOW 2015, Ivanyos et al.,2015)

There is a deterministic polynomial time algorithm to compute the $\mathsf{ncrk}(\mathcal{B})$ *for any matrix space* $\mathcal{B} \leq \mathbb{F}^{n \times n}$.

Approximating the Commutative Rank

Theorem (GGOW 2015, Ivanyos et al.,2015)

There is a deterministic polynomial time algorithm to compute the $\text{ncrk}(\mathcal{B})$ *for any matrix space* $\mathcal{B} \leq \mathbb{F}^{n \times n}$.

- Implies a deterministic polynomial time algorithm computing a $\frac{1}{2}$-approximation of the commutative rank.

Approximating the Commutative Rank

Theorem (GGOW 2015, Ivanyos et al.,2015)

There is a deterministic polynomial time algorithm to compute the $\text{ncrk}(\mathcal{B})$ *for any matrix space* $\mathcal{B} \leq \mathbb{F}^{n \times n}$.

- Implies a deterministic polynomial time algorithm computing a $\frac{1}{2}$-approximation of the commutative rank.
- Improve the approximation ratio?

Outline

Main Contribution

- A deterministic PTAS for computing the Commutative rank.

Main Contribution

- A deterministic PTAS for computing the Commutative rank.

Theorem

For any Matrix space $\mathcal{B} \leq \mathbb{F}^{n \times n}$, a deterministic polynomial time algorithm which outputs a matrix $A \in \mathcal{B}$ with:

$$\text{rank}(A) \geq (1-\epsilon)\text{crk}(\mathcal{B}).$$

Algorithm runs in time $n^{O\left(\frac{1}{\epsilon}\right)}$.

Main Idea

- Define the notion of **Wong Index** $w(A, \mathcal{B})$ for any $A \in \mathcal{B}$.
- If $w(A, \mathcal{B})$ is "high" then $\text{rank}(A)$ is already a good approximation of $\text{crk}(\mathcal{B})$.
  - In fact, we showed this connection even for the non-commutative rank.

Main Idea

- Define the notion of **Wong Index** $w(A, \mathcal{B})$ for any $A \in \mathcal{B}$.
- If $w(A, \mathcal{B})$ is "high" then $\text{rank}(A)$ is already a good approximation of $\text{crk}(\mathcal{B})$.
  - In fact, we showed this connection even for the non-commutative rank.
- If $w(A, \mathcal{B})$ is "low" then in deterministic $n^{O(\frac{1}{\epsilon})}$ time, find a matrix $A' \in \mathcal{B}$ such that $\text{rank}(A') > \text{rank}(A)$.

A min-max characterization of ranks

Theorem

For all matrix spaces $\mathcal{A} = \langle A_1, A_2, \ldots, A_m \rangle \leq \mathbb{F}^{n \times n}$, *we have:*

$$\operatorname{ncrk}(\mathcal{A}) = \min_{B=\{b_1,b_2,\ldots,b_n\} \text{ basis of } \mathbb{F}^n} \; \max_{C_1,C_2,\ldots,C_n \in \mathcal{A}} \operatorname{rank}([C_i b_i]).$$

A min-max characterization of ranks

Theorem

For all matrix spaces $\mathcal{A} = \langle A_1, A_2, \ldots, A_m \rangle \leq \mathbb{F}^{n \times n}$, *we have:*

$$\mathrm{ncrk}(\mathcal{A}) = \min_{B=\{b_1,b_2,\ldots,b_n\} \text{ basis of } \mathbb{F}^n} \; \max_{C_1,C_2,\ldots,C_n \in \mathcal{A}} \mathrm{rank}([C_i b_i]).$$

$$\mathrm{crk}(\mathcal{A}) = \max_{C_1,C_2,\ldots,C_n \in \mathcal{A}} \; \min_{B=\{b_1,b_2,\ldots,b_n\} \text{ basis of } \mathbb{F}^n} \mathrm{rank}([C_i b_i]).$$

Outline

Based on

- Joint work with Prof. Dr. Michael Sagraloff.
- Publications:
  - *Efficiently Computing Real Roots of Sparse Polynomials* Jindal, Gorav, and Sagraloff, Michael **In Proceedings of the 2017 ACM on International Symposium on Symbolic and Algebraic Computation** 2017.

Roots of Polynomials

- We have a degree n (real) polynomial:

$$f(x) = \sum_{i=0}^{n} f_i x^i.$$

- Want to compute its (real) roots.
- In practice, the polynomial f is often "sparse".

Sparse Polynomials

- A polynomial is k-sparse if it has only k non-zero terms.

Sparse Polynomials

- A polynomial is k-sparse if it has only k non-zero terms.

Definition ((n, k, τ)-nomial)

A real polynomial $f(x) \in \mathbb{R}[x]$ is an (n, k, τ)-nomial if:

$$f(x) = \sum_{i=1}^{k} f_i x^{e_i}.$$

Here $0 \leq e_1 < e_2 < \cdots < e_k \leq n$ and $2^{-\tau} \leq |f_i| \leq 2^{\tau}$.

Sparse Polynomials Real Roots

- If $f(x) = \sum_{i=1}^{k} f_i x^{e_i}$, then:

$\text{var}(f) \overset{\text{def}}{=\!=}$ Number of signs changes in the sequence $(f_1, f_2, \ldots, f_k)$.

$N_+(f) \overset{\text{def}}{=\!=}$ Number of positive real roots of f.

Sparse Polynomials Real Roots

- If $f(x) = \sum_{i=1}^{k} f_i x^{e_i}$, then:

$\text{var}(f) \overset{\text{def}}{=\!=}$ Number of signs changes in the sequence $(f_1, f_2, \ldots, f_k)$.

$N_+(f) \overset{\text{def}}{=\!=}$ Number of positive real roots of f.

Theorem (Descartes's rule of signs)

For all $f(x) \in \mathbb{R}[x]$, $\text{var}(f) - N_+(f)$ is a non-negative even integer.

Computing Real Roots of Sparse Polynomials

- Descartes's rule of signs implies that any (n, k, τ)-nomial has at most $2k-1$ real roots.
- For integer (n, k, τ)-nomials, the input size is $O(k(\tau + \log n))$.

Computing Real Roots of Sparse Polynomials

- Descartes's rule of signs implies that any (n, k, τ)-nomial has at most $2k - 1$ real roots.
- For integer (n, k, τ)-nomials, the input size is $O(k(\tau + \log n))$.
- We want to "compute" all the real roots of (n, k, τ)-nomials in time $\text{poly}(k, \tau, \log n)$ ($\#$ bit operations).

Computing Real Roots of Sparse Polynomials

- Descartes's rule of signs implies that any (n, k, τ)-nomial has at most $2k - 1$ real roots.
- For integer (n, k, τ)-nomials, the input size is $O(k(\tau + \log n))$.
- We want to "compute" all the real roots of (n, k, τ)-nomials in time $\text{poly}(k, \tau, \log n)$ (# bit operations).
- "Compute" means to find disjoint and (small) real intervals such that each interval contains exactly one real root (isolating the real roots).

Mignotte Polynomials

- Mignotte polynomial $f(x) = x^n - (2^{2\tau}x^2 - 1)^2$ is a $(n, 4, 4\tau)$-nomial.

Mignotte Polynomials

- Mignotte polynomial $f(x) = x^n - (2^{2\tau}x^2 - 1)^2$ is a $(n, 4, 4\tau)$-nomial.
- It can be shown that f has two real roots in $(a - r, a + r)$ for $a = 2^{-\tau}$ and $r = (2^{1-\tau})^{\frac{n}{2}}$.
  - Two very close real roots and hence hard to isolate them for any efficient algorithm.

Mignotte Polynomials

- Mignotte polynomial $f(x) = x^n - (2^{2\tau}x^2 - 1)^2$ is a $(n, 4, 4\tau)$-nomial.
- It can be shown that f has two real roots in $(a - r, a + r)$ for $a = 2^{-\tau}$ and $r = (2^{1-\tau})^{\frac{n}{2}}$.
  - Two very close real roots and hence hard to isolate them for any efficient algorithm.

Theorem

Any algorithm which isolates the real roots of $f(x) = x^n - (2^{2\tau}x^2 - 1)^2$ requires $\Omega(n\tau)$ bit operations.

Outline

Computing Real Roots of Polynomials

- For $k = n$ (dense case), $\text{poly}(n, \tau)$ time algorithms exist.
  - Pan (2001), Sagraloff, Mehlhorn (2015), Eigenwillig (2006) and many others.

Computing Real Roots of Polynomials

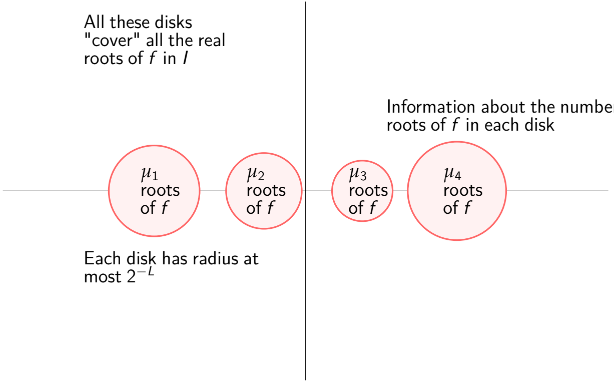
- For $k = n$ (dense case), $\text{poly}(n, \tau)$ time algorithms exist.
  - Pan (2001), Sagraloff, Mehlhorn (2015), Eigenwillig (2006) and many others.
- Integer (n, k, τ)-nomials.
  - Poly time algorithms for isolating integer and rational roots (Cucker et.al, Lenstra, 99).
  - Algorithm to isolate real roots using $\text{poly}(k \cdot (\log n + \tau))$ arithmetic operations. Bit operations still $\tilde{O}(n\tau)$ (Sagraloff (2014)).

Outline

Covering

Definition ((L, I)-covering)

$f \in \mathbb{R}[x], L \in \mathbb{N}, I \subseteq \mathbb{R}$.



Main Result

Theorem

For any (n, k, τ)-nomial, we can compute an L-covering $\mathcal{L}$ of size at most $2k$ in time $\tilde{O}(\text{poly}(k, \log n) \cdot (\tau + L))$.

Main Result

Theorem

For any (n, k, τ)-nomial, we can compute an L-covering $\mathcal{L}$ of size at most $2k$ in time $\tilde{O}(\text{poly}(k, \log n) \cdot (\tau + L))$.

Corollary

If f is an (n, k, τ)-nomial with only simple real roots, and σ is the minimal distance between any two (complex) distinct roots of f, then we can "compute" all the real roots of f in $\tilde{O}\left(\text{poly}(k, \log n)(\tau + \overline{\log}\left(\frac{1}{\sigma}\right))\right)$ bit operations.

Trinomial Root Separation

Theorem (Also proved independently by Koiran)

$f(x) = a_1 x^{e_1} + a_2 x^{e_2} + a_3$ *an integer trinomial with:* $\log \max(e_1, e_2, |a_1|, |a_2|, |a_3|) \leq \tau$. *If* z_1 *and* z_2 *are two distinct roots of* $f(x)$ *then* $|z_1 - z_2| \geq 2^{-c\tau^3}$ *for some* $c < 2^{68}$.

Trinomial Root Separation

Theorem (Also proved independently by Koiran)

$f(x) = a_1 x^{e_1} + a_2 x^{e_2} + a_3$ an integer trinomial with: $\log \max(e_1, e_2, |a_1|, |a_2|, |a_3|) \leq \tau$. If z_1 and z_2 are two distinct roots of $f(x)$ then $|z_1 - z_2| \geq 2^{-c\tau^3}$ for some $c < 2^{68}$.

Corollary

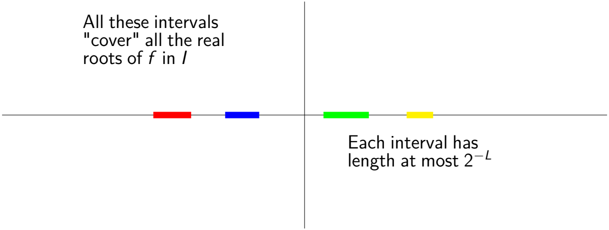
We can isolate all the real roots of trinomials in $\tilde{O}\left(\text{poly}(k, \log n) \cdot \tau^3\right)$ bit operations.

Outline

Weak Covering

Definition

A *weak* (L, I)*-covering* for f is a list $(I_1, I_2, \ldots, I_t)$ of disjoint and sorted real intervals:



T_ℓ-Test

Polynomial $F \in \mathbb{C}[x]$, a disk $\Delta = \Delta_r(m) \subset \mathbb{C}$, and $K \geq 1$, define T_ℓ-Test:

$$T_\ell(\Delta, K, F) : \left| \frac{F^{(\ell)}(m) r^\ell}{\ell!} \right| - K \cdot \sum_{i \neq \ell} \left| \frac{F^{(i)}(m) r^i}{i!} \right| > 0.$$

If T_ℓ-Test succeeds for any $K \geq 1$, then Δ contains exactly ℓ roots of F counted with multiplicity.

T_ℓ-Test

Polynomial $F \in \mathbb{C}[x]$, a disk $\Delta = \Delta_r(m) \subset \mathbb{C}$, and $K \geq 1$, define T_ℓ-Test:

$$T_\ell(\Delta, K, F) : \left|\frac{F^{(\ell)}(m) r^\ell}{\ell!}\right| - K \cdot \sum_{i \neq \ell} \left|\frac{F^{(i)}(m) r^i}{i!}\right| > 0.$$

If T_ℓ-Test succeeds for any $K \geq 1$, then Δ contains exactly ℓ roots of F counted with multiplicity.

Theorem (Becker, Sagraloff, Sharma, Yap 2018)

If both Δ and Δ' contain ℓ roots with $\Delta \subseteq \Delta'$ and Δ' being sufficiently large, then T_ℓ-Test succeeds on some disk D with $\Delta \subseteq D \subseteq \Delta'$.

Main Algorithm

1: Compute a weak $(L', [0,1])$-covering $\mathcal{L}$ for f that is "well-separated".
2: **for** each interval $I \in \mathcal{L}$ **do**
3: $\quad \Delta \leftarrow$ Disk whose diameter is I

Main Algorithm

1: Compute a weak $(L', [0,1])$-covering $\mathcal{L}$ for f that is "well-separated".
2: **for** each interval $I \in \mathcal{L}$ **do**
3: $\Delta \leftarrow$ Disk whose diameter is I
4: Using T_ℓ-Test, count number of roots $\mu_{\Delta'}$ in a super disk Δ' of Δ.
5: Output $(\Delta', \mu_{\Delta'})$.
6: **end for**

Computing a Weak Covering

- Suppose we already have a covering W' for f'.

```
for each consecutive intervals (a, b) and (c, d) in W' do
    Compute signs of f(b) and f(c).
    if f(b)f(c) < 0 then
```

Computing a Weak Covering

- Suppose we already have a covering W' for f'.

```
for each consecutive intervals (a, b) and (c, d) in W' do
    Compute signs of f(b) and f(c).
    if f(b)f(c) < 0 then
        Refine the isolating interval (b, c) to a new interval (b', c') of desired length.
        Add (b', c')
    end if
end for
Also add intervals of W'.
```

Challenges

- Computing the sign of f at end points.

Challenges

- Computing the sign of f at end points.
- Refining an interval to a small length.
- T_ℓ-Test
  - How to make sure it succeeds?
  - Adapting it to the sparse case.

Based on

- Joint work with Prof. Dr. Markus Bläser.
- Publications:
  - *On the Complexity of Symmetric Polynomials* Bläser, Markus, and Jindal, Gorav In **10th Innovations in Theoretical Computer Science Conference (ITCS)** 2019.

Outline

Symmetric Polynomial Complexity

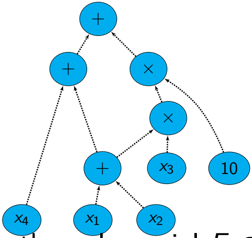
- Any symmetric Boolean function $f : \{0,1\}^n \to \{0,1\}$ is "easy" to compute.

Symmetric Polynomial Complexity

- Any symmetric Boolean function $f : \{0,1\}^n \rightarrow \{0,1\}$ is "easy" to compute.
- Lipton and Regan (Gödel's Lost Letter and P = NP, 2009) ask:
  - Are symmetric polynomials (families) also "easy" to compute?

Polynomials and Arithmetic Circuits

- Every arithmetic circuit computes a polynomial and vice versa.



- Above circuit computes the polynomial $F \in \mathbb{C}[x_1, x_2, x_3, x_4]$ where $F = 10x_3(x_1 + x_2) + x_1 + x_2 + x_4$.
  - Size and depth have same definitions as in the Boolean case.

Arithmetic Complexity

Definition

The arithmetic complexity $L(f)$ of a polynomial $f \in \mathbb{C}[x_1, x_2, \ldots, x_n]$ is defined as the minimum size of any arithmetic circuit computing F.

- Thus $L(F) \leq 10$, where $F = 10x_3(x_1 + x_2) + x_1 + x_2 + x_4$.

Fundamental Theorem

Theorem (Fundamental Theorem of Symmetric Polynomials)

If $g \in \mathbb{C}[x_1, x_2, \ldots, x_n]$ is a symmetric polynomial, then there is a unique $f \in \mathbb{C}[y_1, y_2, \ldots, y_n]$ such that $g = f(e_1, e_2, \ldots, e_n)$. Here e_i's elementary symmetric polynomials.

Fundamental Theorem

Theorem (Fundamental Theorem of Symmetric Polynomials)

If $g \in \mathbb{C}[x_1, x_2, \ldots, x_n]$ is a symmetric polynomial, then there is a unique $f \in \mathbb{C}[y_1, y_2, \ldots, y_n]$ such that $g = f(e_1, e_2, \ldots, e_n)$. Here e_i's elementary symmetric polynomials.

- Write symmetric polynomials always with f_{Sym}. Hence the bijection $f(e_1, e_2, \ldots, e_n) = f_{\mathsf{Sym}}$:

$$f \iff f_{\mathsf{Sym}}.$$

Fundamental Theorem

Theorem (Fundamental Theorem of Symmetric Polynomials)

If $g \in \mathbb{C}[x_1, x_2, \ldots, x_n]$ is a symmetric polynomial, then there is a unique $f \in \mathbb{C}[y_1, y_2, \ldots, y_n]$ such that $g = f(e_1, e_2, \ldots, e_n)$. Here e_i's elementary symmetric polynomials.

- Write symmetric polynomials always with f_{Sym}. Hence the bijection $f(e_1, e_2, \ldots, e_n) = f_{\text{Sym}}$:

$$f \iff f_{\text{Sym}}.$$

Idea

Study the connection between $L(f)$ and $L(f_{\text{Sym}})$.

Relation between $L(f)$ and $L(f_{\mathrm{Sym}})$

Lemma

For all $f \in \mathbb{C}[x_1, x_2, \ldots, x_n]$, $L(f_{\mathrm{Sym}}) \leq L(f) + O(n^2)$.

Proof.

Replace x_i by e_i, e_i's can be computed a circuit of size $O(n^2)$.

Relation between $L(f)$ and $L(f_{\mathrm{Sym}})$

Lemma

For all $f \in \mathbb{C}[x_1, x_2, \ldots, x_n]$, $L(f_{\mathrm{Sym}}) \leq L(f) + O(n^2)$.

Proof.

Replace x_i by e_i, e_i's can be computed a circuit of size $O(n^2)$. □

- Can we also bound $L(f)$ polynomially in terms of $L(f_{\mathrm{Sym}})$?
  - Lipton and Regan (Gödel's Lost Letter and P = NP, 2009) ask this question.

Outline

Main Theorem

Theorem

For any polynomial $f \in \mathbb{C}[x_1, x_2, \ldots, x_n]$ *of degree* d*,*
$L(f) \leq \tilde{O}\left(d^2 L(f_{\mathrm{Sym}}) + d^2 n^2\right)$.

- Previous best bound: $L(f) \leq 4^n (n!)^2 (L(f_{\mathrm{Sym}}) + 2)$.

Main Theorem

Theorem

For any polynomial $f \in \mathbb{C}[x_1, x_2, \ldots, x_n]$ *of degree* d,
$L(f) \leq \tilde{O}\left(d^2 L(f_{\mathsf{Sym}}) + d^2 n^2\right)$.

- Previous best bound: $L(f) \leq 4^n (n!)^2 (L(f_{\mathsf{Sym}}) + 2)$.

Corollary

Assuming VP $\neq$ VNP, *symmetric polynomial family* $(q_n)_{n \in \mathbb{N}}$ *defined by* $q_n \stackrel{def}{=\!=} (\mathrm{per}_n)_{\mathsf{Sym}}$ *has super polynomial arithmetic complexity.*

Checking Symmetries

Theorem

Checking if a given Boolean function is symmetric is as hard as CSAT.

Checking Symmetries

Theorem

Checking if a given Boolean function is symmetric is as hard as CSAT.

Theorem

Checking if a given polynomial is symmetric is as hard as PIT.

Thanks

Thank you for your attention!

Additional Material

- Non-commutative rank definition
- Alternative Proof of PTAS

Additional Material

- Rouché's Theorem
- Pellet's Theorem

Additional Material

- Symmetric Boolean functions
- Algebraic Complexity Theory
- Symmetric and elementary symmetric polynomials
- Idea for proof of $L(f) \leq \tilde{O}\left(d^2 L(f_{\mathrm{Sym}}) + d^2 n^2\right)$

Non-commutative rank

- (c-shrunk subspace) $V \leq \mathbb{F}^n$ is a c-shrunk subspace of $\mathcal{B} \leq \mathbb{F}^{n \times n}$, if $\dim(\mathcal{B}V) \leq \dim(V) - c$.

Definition (Non-commutative rank)

For any matrix space $\mathcal{B} \leq \mathbb{F}^{n \times n}$, if
$r = \max\{c \mid \exists\, c\text{-shrunk subspaceof } \mathcal{B}\}$ then
Non-commutaive rank of $\mathcal{B} = \text{ncrk}(\mathcal{B}) = n - r$. Go Back

Outline

Main Idea

- $\mathcal{B} = \langle B_1, B_2, \ldots, B_m \rangle \leq \mathbb{F}^{n \times n}$.
  - $B = x_1 B_1 + x_2 B_2 + \ldots + x_m B_m$ over the field $\mathbb{F}(x_1, x_2, \ldots, x_m)$.
- We have some $A \in \mathcal{B}$ with some rank r.
  - Want to find $A' \in \mathcal{B}$ with $\text{rank}(A') > r$.
- WLOG assume $A = \begin{bmatrix} I_r & 0 & \ldots & 0 \\ 0 & 0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \ldots & 0 & 0 \end{bmatrix}$.
- Consider the matrix $A + B \in \mathbb{F}(x_1, x_2, \ldots, x_m)^{n \times n}$. Go Back

Main idea (Continued)

- $A + B = \begin{bmatrix} I_r + B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$.
- Suppose $B_{22} = 0$ then $\text{rank}(A + B) = \text{rank}(B) \leq 2r$.
  - rank(A) is already $\frac{1}{2}$-approximation of rank(B).
- Otherwise $B_{22} \neq 0$, $c(x_1, x_2, \ldots, x_m)$ be a non-zero entry of B_{22}. Go Back

Main idea (Continued)

- Consider the Minor M of $A+B$ which has $c(x_1, x_2, \dots, x_m)$ as the last entry.
  - $M = \begin{bmatrix} 1+\ell_{11} & \ell_{12} & \dots & a_1 \\ \ell_{21} & 1+\ell_{22} & \dots & a_2 \\ \vdots & \vdots & \ddots & \vdots \\ b_1 & b_2 & \dots & c(x_1, x_2, \dots, x_m) \end{bmatrix}_{(r+1)\times(r+1)}$
- $\det(M(x_1, x_2, \dots, x_m)) = c(x_1, x_2, \dots, x_m) +$ terms of degree at least 2.
  - Thus easy PIT for $\det(M(x_1, x_2, \dots, x_m))$ and hence rank increase. Go Back

Outline

Rouché's Theorem

Theorem (Rouché's Theorem)

Let f and g be holomorphic inside some region Δ with boundary $\partial\Delta$. If $|f(z)| > |f(z) - g(z)|$ on $\partial\Delta$, then f and g have the same number of zeros inside Δ. Go Back

Pellet's Theorem

Theorem (Pellet's Theorem)

Given the polynomial

$$f(z) = f_0 + f_1 x + \cdots + f_p x^p + \cdots + f_n x^n \quad \textit{with } f_p \neq 0.$$

If the polynomial $F_p(x)$ defined by

$$F_p(x) \stackrel{def}{=\!=} |f_0| + |f_1| x + \cdots + |f_{p-1}| x^p - |f_p| x^p + |f_{p+1}| x^p + \cdots + |f_n| x^n$$

has two positive zeros r and R, $r < R$, then $f(x)$ has exactly p zeros in or on the circle $|x| < r$ and no zeros in the ring $r < |x| < R$. Go Back

Outline

Symmetric Boolean Functions

Definition

A Boolean function $f : \{0,1\}^n \rightarrow \{0,1\}$ is said to be symmetric if it is invariant under any permutation of its inputs.

- Can a symmetric Boolean function be hard to compute?

Fact

A symmetric Boolean function only depends on the number of 1's in the input and thus can be computed by constant depth threshold circuits (complexity class TC^0*). Therefore "easy" to compute.* Go Back

Hard Polynomial families

Goal

Find polynomial families $\{f_1, f_2, \ldots, f_n, \ldots,\}$ such that $L(f_n)$ is a super polynomial function of n.

- The permanent family defined by $\text{per}_n \stackrel{\text{def}}{=\!=} \sum_{\pi \in \mathfrak{S}_n} \prod_{i=1}^{n} x_{i,\pi(i)}$ is believed to be "hard".
  - Known as VP vs VNP conjecture. Go Back

Outline

Symmetric Polynomials

Definition

A polynomial $f \in \mathbb{C}[x_1, x_2, \ldots, x_n]$ is said to be symmetric if it is invariant under any permutation of its inputs.

Example

$x_1^2 + x_2^2 + x_1x_2 \in \mathbb{C}[x_1, x_2]$ is symmetric whereas $x_1^2 + x_2$ is not.

Question

Lipton and Regan (Gödel's Lost Letter and P = NP, 2009) ask whether we can find hard (families of) symmetric polynomials?

Go Back

Elementary Symmetric Polynomials

Definition

The i^{th} elementary symmetric polynomial e_i in n variables $x_1, x_2, \ldots,$ x_n is defined as:

$$e_i \stackrel{\text{def}}{=\!=} \sum_{1 \le j_1 < j_2 < \cdots < j_i \le n} x_{j_1} \cdot x_{j_2} \cdot \cdots \cdot x_{j_i}.$$

- e_i's are obviously symmetric.
- Sum and product of symmetric polynomials is also symmetric.
- Thus the polynomials in the algebra generated by e_i's are also symmetric. Lipton and Regan (Gödel's Lost Letter and P = NP, 2009) ask whether we can find hard (families of) symmetric polynomials? Go Back

Main idea

Example

Suppose $f_{\mathrm{Sym}} = x_1^2 + x_2^2 + x_1x_2 = e_1^2 - e_2$. Given an arithmetic circuit for f_{Sym}, we want to get a circuit for $f = e_1^2 - e_2$.

Idea

x_1, x_2 are the roots of polynomial:
$B(y) \stackrel{\mathrm{def}}{=\!=} y^2 - (x_1 + x_2)y + x_1x_2 = y^2 - e_1y + e_2$. Thus:

$$x_1 = \frac{e_1 + \sqrt{e_1^2 - 4e_2}}{2}. \quad (1)$$

$$x_2 = \frac{e_1 - \sqrt{e_1^2 - 4e_2}}{2}. \quad (2)$$

Main idea (Continued)

- If we substitute:

$$x_1 = \frac{e_1 + \sqrt{e_1^2 - 4e_2}}{2}. \tag{3}$$

$$x_2 = \frac{e_1 - \sqrt{e_1^2 - 4e_2}}{2}. \tag{4}$$

in the circuit for f_{Sym}, we obtain a circuit for f. How to compute the above radical expressions?

- These are not even polynomials. Go Back

Main idea (Continued)

- Use the substitution $e_2 \leftarrow e_2 - 1$ and then substitute x_1 and x_2 in $f_{\mathrm{Sym}}(x_1, x_2)$ to obtain $f(e_1, e_2 - 1)$.
  - But even after this $e_2 \leftarrow e_2 - 1$, radical expressions for x_1, x_2 are not polynomials.
- But they are power series (use Taylor expansion).
  - We can not compute power series using arithmetic circuits.

Idea

Only need to compute degree two truncations of these power series, because f is of degree two. Go Back