Determinant Versus Permanent

Manindra Agrawal

IIT Kanpur

ICM 2006

Overview

1. Determinant and Permanent
2. Complexity Notions
3. Known Lower Bounds on Complexity of Permanent
4. Proving Strong Lower Bounds on Determinant Complexity
5. Proving Strong Lower Bounds on Circuit Complexity
6. Proving Hardness of Permanent Polynomial

Outline

1. Determinant and Permanent
2. Complexity Notions
3. Known Lower Bounds on Complexity of Permanent
4. Proving Strong Lower Bounds on Determinant Complexity
5. Proving Strong Lower Bounds on Circuit Complexity
6. Proving Hardness of Permanent Polynomial

Determinant

Determinant of an $n \times n$ matrix $X = [x_{i,j}]$ is defined as:

$$\det X = \sum_{\sigma \in S_n} \mathrm{sgn}(\sigma) \cdot \prod_{i=1}^{n} x_{i,\sigma(i)}.$$

Here S_n is the group of all permutations on $[1, n]$ and $\mathrm{sgn}(\sigma)$ is the sign of the permulation σ, $\mathrm{sgn}(\sigma) \in \{1, -1\}$.

Properties of Determinant

Linearity. $\det[c_1 + c_1' \; c_2 \; \cdots \; c_n] = \det[c_1 \; c_2 \; \cdots \; c_n] + \det[c_1' \; c_2 \; \cdots \; c_n]$.

Multiplicativity. $\det AB = \det A \cdot \det B$.

Geometric Interpretation. $|\det[c_1 \; c_2 \; \cdots \; c_n]|$ is the volume of the parallelopied defined by vectors c_1, c_2, ..., c_n.

Algebraic Interpretation. $\det A = \prod_{i=1}^{n} \lambda_i$ where $\lambda_1, \ldots, \lambda_n$ are eigenvalues of A.

Relation to Multiplication. For any A, there exists an efficiently computable B and number m such that $\det A = [B^m]_{1,1}$.

Properties of Determinant

Linearity. $\det[c_1 + c'_1 \; c_2 \; \cdots \; c_n] = \det[c_1 \; c_2 \; \cdots \; c_n] + \det[c'_1 \; c_2 \; \cdots \; c_n]$.

Multiplicativity. $\det AB = \det A \cdot \det B$.

Geometric Interpretation. $|\det[c_1 \; c_2 \; \cdots \; c_n]|$ is the volume of the parallelopied defined by vectors $c_1, c_2, \ldots, c_n$.

Algebraic Interpretation. $\det A = \prod_{i=1}^{n} \lambda_i$ where $\lambda_1, \ldots, \lambda_n$ are eigenvalues of A.

Relation to Multiplication. For any A, there exists an efficiently computable B and number m such that $\det A = [B^m]_{1,1}$.

Properties of Determinant

Linearity. $\det[c_1 + c_1' \; c_2 \; \cdots \; c_n] = \det[c_1 \; c_2 \; \cdots \; c_n] + \det[c_1' \; c_2 \; \cdots \; c_n]$.

Multiplicativity. $\det AB = \det A \cdot \det B$.

Geometric Interpretation. $|\det[c_1 \; c_2 \; \cdots \; c_n]|$ is the volume of the parallelopied defined by vectors c_1, c_2, …, c_n.

Algebraic Interpretation. $\det A = \prod_{i=1}^{n} \lambda_i$ where $\lambda_1, \ldots, \lambda_n$ are eigenvalues of A.

Relation to Multiplication. For any A, there exists an efficiently computable B and number m such that $\det A = [B^m]_{1,1}$.

Properties of Determinant

LINEARITY. $\det[c_1 + c_1' \; c_2 \; \cdots \; c_n] = \det[c_1 \; c_2 \; \cdots \; c_n] + \det[c_1' \; c_2 \; \cdots \; c_n]$.

MULTIPLICATIVITY. $\det AB = \det A \cdot \det B$.

GEOMETRIC INTERPRETATION. $|\det[c_1 \; c_2 \; \cdots \; c_n]|$ is the volume of the parallelopied defined by vectors $c_1, c_2, \ldots, c_n$.

ALGEBRAIC INTERPRETATION. $\det A = \prod_{i=1}^{n} \lambda_i$ where $\lambda_1, \ldots, \lambda_n$ are eigenvalues of A.

RELATION TO MULTIPLICATION. For any A, there exists an efficiently computable B and number m such that $\det A = [B^m]_{1,1}$.

Properties of Determinant

Linearity. $\det[c_1 + c'_1 \; c_2 \; \cdots \; c_n] = \det[c_1 \; c_2 \; \cdots \; c_n] + \det[c'_1 \; c_2 \; \cdots \; c_n]$.

Multiplicativity. $\det AB = \det A \cdot \det B$.

Geometric Interpretation. $|\det[c_1 \; c_2 \; \cdots \; c_n]|$ is the volume of the parallelopied defined by vectors c_1, c_2, …, c_n.

Algebraic Interpretation. $\det A = \prod_{i=1}^{n} \lambda_i$ where $\lambda_1, \ldots, \lambda_n$ are eigenvalues of A.

Relation to Multiplication. For any A, there exists an efficiently computable B and number m such that $\det A = [B^m]_{1,1}$.

Permanent

Permanent of an $n \times n$ matrix $X = [x_{i,j}]$ is defined as:

$$\text{per } X = \sum_{\sigma \in S_n} \prod_{i=1}^{n} x_{i,\sigma(i)}.$$

Same as determinant except the signs.

Permanent

Permanent of an $n \times n$ matrix $X = [x_{i,j}]$ is defined as:

$$\text{per } X = \sum_{\sigma \in S_n} \prod_{i=1}^{n} x_{i,\sigma(i)}.$$

Same as determinant except the signs.

Properties of Permanent

Linearity. $\text{per}\,[c_1 + c_1'\ c_2\ \cdots\ c_n] =$
$\text{per}\,[c_1\ c_2\ \cdots\ c_n] + \text{per}\,[c_1'\ c_2\ \cdots\ c_n]$.

Combinatorial Interpretation. Permanent of matrix A with non-negative numbers is the sum of weights of all perfect matchings of the bipartite graph represented by A.

Properties of Permanent

Linearity. $\text{per}\,[c_1 + c_1' \; c_2 \; \cdots \; c_n] =$
$\text{per}\,[c_1 \; c_2 \; \cdots \; c_n] + \text{per}\,[c_1' \; c_2 \; \cdots \; c_n]$.

Combinatorial Interpretation. Permanent of matrix A with non-negative numbers is the sum of weights of all perfect matchings of the bipartite graph represented by A.

Properties of Permanent

- Despite closeness in definition, permanent function satisfies much fewer properties than determinant function.
- In particular, there are efficient algorithms to compute determinant, however, there does not appear any way of computing permanent efficiently.

Properties of Permanent

- Despite closeness in definition, permanent function satisfies much fewer properties than determinant function.
- In particular, there are efficient algorithms to compute determinant, however, there does not appear any way of computing permanent efficiently.

Computational Characterizations

Valiant (1979) showed that

- The complexity of computing permanent polynomial charecterizes arithmetic version of NP.
- The complexity of computing determinant polynomial (nearly) charecterizes arithmetic version of P.

Computational Characterizations

Valiant (1979) showed that

- The complexity of computing permanent polynomial charecterizes arithmetic version of NP.
- The complexity of computing determinant polynomial (nearly) charecterizes arithmetic version of P.

Computing Determinant and Permanent

Intuition. Permanent is much harder to compute than determinant.

This can be formalized in two ways:

- Permanent of X has a large determinant complexity.
- Permanent of X has large circuit complexity.

Computing Determinant and Permanent

Intuition. Permanent is much harder to compute than determinant.

This can be formalized in two ways:

- Permanent of X has a large determinant complexity.
- Permanent of X has large circuit complexity.

Computing Determinant and Permanent

Intuition. Permanent is much harder to compute than determinant.

This can be formalized in two ways:

- Permanent of X has a large determinant complexity.
- Permanent of X has large circuit complexity.

Outline

1. Determinant and Permanent
2. Complexity Notions
3. Known Lower Bounds on Complexity of Permanent
4. Proving Strong Lower Bounds on Determinant Complexity
5. Proving Strong Lower Bounds on Circuit Complexity
6. Proving Hardness of Permanent Polynomial

Determinant Complexity

For matrix $X = [x_{i,j}]$, permanent of X has determinant complexity m over field F if there exists an $m \times m$ matrix Y such that

- per $X = \det Y$.
- Each entry of Y is an F-affine combination of $x_{i,j}$'s.

Arithmetic Circuits

Arithmetic circuits over field F represent a sequence of arithmetic operations over F on variables.

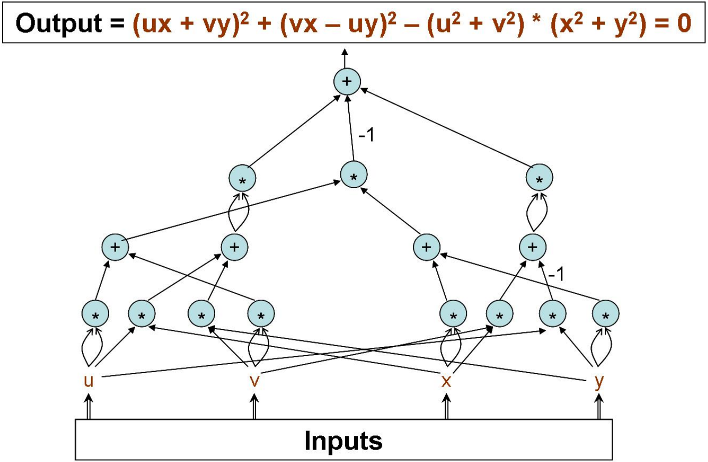
- The variables are called input to the circuit.
- The result of the operations is called the output of the circuit.

ARITHMETIC CIRCUITS

Arithmetic circuits over field F represent a sequence of arithmetic operations over F on variables.

- The variables are called input to the circuit.
- The result of the operations is called the output of the circuit.

Arithmetic Circuits



Circuit Complexity

Crucial parameters associated with arithmetic circuits are:

- Size: equals the number of operations in the circuit.
- Depth: equals the length of the longest path from a variable to output of the circuit.
- Degree: equals the formal degree of the polynomial output by the circuit.

Circuit complexity of a polynomial is the size of the smallest arithmetic circuit that outputs the polynomial.

Circuit Complexity

Crucial parameters associated with arithmetic circuits are:

- Size: equals the number of operations in the circuit.
- Depth: equals the length of the longest path from a variable to output of the circuit.
- Degree: equals the formal degree of the polynomial output by the circuit.

Circuit complexity of a polynomial is the size of the smallest arithmetic circuit that outputs the polynomial.

Circuit Complexity

Crucial parameters associated with arithmetic circuits are:

- Size: equals the number of operations in the circuit.
- Depth: equals the length of the longest path from a variable to output of the circuit.
- Degree: equals the formal degree of the polynomial output by the circuit.

Circuit complexity of a polynomial is the size of the smallest arithmetic circuit that outputs the polynomial.

Circuit Complexity

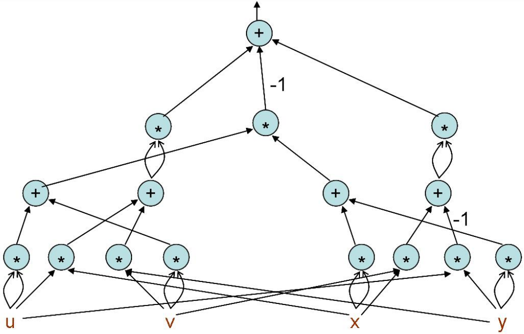
Crucial parameters associated with arithmetic circuits are:

- Size: equals the number of operations in the circuit.
- Depth: equals the length of the longest path from a variable to output of the circuit.
- Degree: equals the formal degree of the polynomial output by the circuit.

Circuit complexity of a polynomial is the size of the smallest arithmetic circuit that outputs the polynomial.

Circuit Complexity

Output = $(ux + vy)^2 + (vx - uy)^2 - (u^2 + v^2) * (x^2 + y^2) = 0$



Size = 16 | **Depth = 4** | **Degree = 4**

arith-P and arith-NP

Polynomial family $\{p_n\} \in$ arith-P if p_n has circuit complexity $n^{O(1)}$.

Polynomial family $\{q_n\} \in$ arith-NP if there exists a family $\{p_n\} \in$ arith-P such that

$$q_n(x_1, \ldots, x_n) = \sum_{y_1=0}^{1} \cdots \sum_{y_n=0}^{1} p_{2n}(x_1, \ldots, x_n, y_1, \ldots, y_n).$$

arith-P and arith-NP

Polynomial family $\{p_n\} \in$ arith-P if p_n has circuit complexity $n^{O(1)}$.

Polynomial family $\{q_n\} \in$ arith-NP if there exists a family $\{p_n\} \in$ arith-P such that

$$q_n(x_1, \ldots, x_n) = \sum_{y_1=0}^{1} \cdots \sum_{y_n=0}^{1} p_{2n}(x_1, \ldots, x_n, y_1, \ldots, y_n).$$

Determinant Complexity versus Circuit Complexity

- Determinant polynomial family over F is in arith-P.
- A polynomial family in arith-P has determinant complexity $n^{O(\log n)}$.

Hypothesis. Permanent of $n \times n$ matrix X over F has superpolynomial circuit complexity for char $F \neq 2$.

Determinant Complexity versus Circuit Complexity

- Determinant polynomial family over F is in arith-P.
- A polynomial family in arith-P has determinant complexity $n^{O(\log n)}$.

Hypothesis. Permanent of $n \times n$ matrix X over F has superpolynomial circuit complexity for char $F \neq 2$.

Determinant Complexity versus Circuit Complexity

- Determinant polynomial family over F is in arith-P.
- A polynomial family in arith-P has determinant complexity $n^{O(\log n)}$.

HYPOTHESIS. Permanent of $n \times n$ matrix X over F has superpolynomial circuit complexity for char $F \neq 2$.

Outline

1. Determinant and Permanent
2. Complexity Notions
3. Known Lower Bounds on Complexity of Permanent
4. Proving Strong Lower Bounds on Determinant Complexity
5. Proving Strong Lower Bounds on Circuit Complexity
6. Proving Hardness of Permanent Polynomial

Lower Bounds for Determinant Complexity

- Mignon and Ressayre (2004) showed that determinant complexity of per X (size $X = n$) is $\Omega(n^2)$ over $\mathbb{Q}$.

Lower Bounds for Circuit Complexity

- Lower bounds are known for permanent only for very restricted type of circuits.
- Jerrum and Snir (1982) showed that any monotone circuit computing per X is of exponential size.
  - Monotone circuits are circuits with no negative constant.
- Shpilka and Wigderson (1999) showed that any depth three circuit computing per X (or even $\det X$) over $\mathbb{Q}$ is of size $\Omega(n^2)$.

Lower Bounds for Circuit Complexity

- Lower bounds are known for permanent only for very restricted type of circuits.
- Jerrum and Snir (1982) showed that any monotone circuit computing per X is of exponential size.
  - Monotone circuits are circuits with no negative constant.
- Shpilka and Wigderson (1999) showed that any depth three circuit computing per X (or even $\det X$) over $\mathbb{Q}$ is of size $\Omega(n^2)$.

Lower Bounds for Circuit Complexity

- Lower bounds are known for permanent only for very restricted type of circuits.
- Jerrum and Snir (1982) showed that any monotone circuit computing $\operatorname{per} X$ is of exponential size.
  - Monotone circuits are circuits with no negative constant.
- Shpilka and Wigderson (1999) showed that any depth three circuit computing $\operatorname{per} X$ (or even $\det X$) over $\mathbb{Q}$ is of size $\Omega(n^2)$.

Lower Bounds for Circuit Complexity

- Grigoriev and Razborov (2000) showed that any depth three circuit computing per X or $\det X$ over a finite field is of exponential size.
- Raz (2004) showed that any multilinear formula computing per X or $\det X$ is of size $n^{\Omega(\log n)}$.
  - Formulas are circuits with outdegree one.
  - Multilinear formulas are formulas in which every gate computes a multilinear polynomial.

Lower Bounds for Circuit Complexity

- Grigoriev and Razborov (2000) showed that any depth three circuit computing per X or $\det X$ over a finite field is of exponential size.
- Raz (2004) showed that any multilinear formula computing per X or $\det X$ is of size $n^{\Omega(\log n)}$.
  - Formulas are circuits with outdegree one.
  - Multilinear formulas are formulas in which every gate computes a multilinear polynomial.

Outline

1. Determinant and Permanent
2. Complexity Notions
3. Known Lower Bounds on Complexity of Permanent
4. Proving Strong Lower Bounds on Determinant Complexity
5. Proving Strong Lower Bounds on Circuit Complexity
6. Proving Hardness of Permanent Polynomial

Geometric Invariant Theory Approach

- Mulmulay and Sohoni (2002) have formulated the problem as an algebraic geometry problem.
- Let $X_\ell = [x_{i,j}]_{1 \le i,j \le \ell}$ be $\ell \times \ell$ matrix of variables.
- Let $\text{per}_\ell = \text{per}\, X_\ell$ and $\det_\ell = \det X_\ell$ denote the permanent and determinant polynomials respectively in ℓ^2 variables.
- Suppose over $\mathbb{Q}$, determinant complexity of per_n is m.
- Let $\text{per}_n = \det Y$ for $m \times m$ matrix Y whose entries are affine combinations of variables of X_n.
- Define $\widehat{\text{per}}_n = x_{m,m}^{m-n} \cdot \text{per}_n$.

Geometric Invariant Theory Approach

- Mulmulay and Sohoni (2002) have formulated the problem as an algebraic geometry problem.
- Let $X_\ell = [x_{i,j}]_{1 \le i,j \le \ell}$ be $\ell \times \ell$ matrix of variables.
- Let $\text{per}_\ell = \text{per}\, X_\ell$ and $\det_\ell = \det X_\ell$ denote the permanent and determinant polynomials respectively in ℓ^2 variables.
- Suppose over $\mathbb{Q}$, determinant complexity of per_n is m.
- Let $\text{per}_n = \det Y$ for $m \times m$ matrix Y whose entries are affine combinations of variables of X_n.
- Define $\widehat{\text{per}}_n = x_{m,m}^{m-n} \cdot \text{per}_n$.

Geometric Invariant Theory Approach

- Mulmulay and Sohoni (2002) have formulated the problem as an algebraic geometry problem.
- Let $X_\ell = [x_{i,j}]_{1 \leq i,j \leq \ell}$ be $\ell \times \ell$ matrix of variables.
- Let $\text{per}_\ell = \text{per}\, X_\ell$ and $\det_\ell = \det X_\ell$ denote the permanent and determinant polynomials respectively in ℓ^2 variables.
- Suppose over $\mathbb{Q}$, determinant complexity of per_n is m.
- Let $\text{per}_n = \det Y$ for $m \times m$ matrix Y whose entries are affine combinations of variables of X_n.
- Define $\widehat{\text{per}}_n = x_{m,m}^{m-n} \cdot \text{per}_n$.

Geometric Invariant Theory Approach

- Mulmulay and Sohoni (2002) have formulated the problem as an algebraic geometry problem.
- Let $X_\ell = [x_{i,j}]_{1 \leq i,j \leq \ell}$ be $\ell \times \ell$ matrix of variables.
- Let $\text{per}_\ell = \text{per } X_\ell$ and $\det_\ell = \det X_\ell$ denote the permanent and determinant polynomials respectively in ℓ^2 variables.
- Suppose over $\mathbb{Q}$, determinant complexity of per_n is m.
- Let $\text{per}_n = \det Y$ for $m \times m$ matrix Y whose entries are affine combinations of variables of X_n.
- Define $\widehat{\text{per}}_n = x_{m,m}^{m-n} \cdot \text{per}_n$.

Geometric Invariant Theory Approach

- This can be expressed as:

$$\widehat{\mathrm{per}}_n = \det{}_m \cdot A$$

where A is a (non-invertible) matrix over $\mathbb{Q}$.

- Let $V = \mathbb{C}^M$ where $M = \binom{m^2+m-1}{m}$ and $P(V)$ be the corresponding projective space.
- Polynomials $\det{}_m$ and $\widehat{\mathrm{per}}_n$ can be viewed as points in $P(V)$.
- Let O be the orbit of $\det{}_m$ under the action of $SL_{m^2}(\mathbb{C})$:

$$O = \{\det{}_m \cdot B \mid B \in SL_{m^2}(\mathbb{C})\}.$$

- It follows that $\widehat{\mathrm{per}}_n$ lies in the closure of O.

Geometric Invariant Theory Approach

- This can be expressed as:

$$\widehat{\text{per}}_n = \det{}_m \cdot A$$

  where A is a (non-invertible) matrix over $\mathbb{Q}$.
- Let $V = \mathbb{C}^M$ where $M = \binom{m^2+m-1}{m}$ and $P(V)$ be the corresponding projective space.
- Polynomials $\det{}_m$ and $\widehat{\text{per}}_n$ can be viewed as points in $P(V)$.
- Let O be the orbit of $\det{}_m$ under the action of $SL_{m^2}(\mathbb{C})$:

$$O = \{\det{}_m \cdot B \mid B \in SL_{m^2}(\mathbb{C})\}.$$

- It follows that $\widehat{\text{per}}_n$ lies in the closure of O.

Geometric Invariant Theory Approach

- This can be expressed as:

$$\widehat{\text{per}}_n = \det{}_m \cdot A$$

where A is a (non-invertible) matrix over $\mathbb{Q}$.

- Let $V = \mathbb{C}^M$ where $M = \binom{m^2+m-1}{m}$ and $P(V)$ be the corresponding projective space.
- Polynomials $\det{}_m$ and $\widehat{\text{per}}_n$ can be viewed as points in $P(V)$.
- Let O be the orbit of $\det{}_m$ under the action of $SL_{m^2}(\mathbb{C})$:

$$O = \{\det{}_m \cdot B \mid B \in SL_{m^2}(\mathbb{C})\}.$$

- It follows that $\widehat{\text{per}}_n$ lies in the closure of O.

Geometric Invariant Theory Approach

- This can be expressed as:

$$\widehat{\text{per}}_n = \det{}_m \cdot A$$

  where A is a (non-invertible) matrix over $\mathbb{Q}$.

- Let $V = \mathbb{C}^M$ where $M = \binom{m^2+m-1}{m}$ and $P(V)$ be the corresponding projective space.
- Polynomials $\det{}_m$ and $\widehat{\text{per}}_n$ can be viewed as points in $P(V)$.
- Let O be the orbit of $\det{}_m$ under the action of $SL_{m^2}(\mathbb{C})$:

$$O = \{\det{}_m \cdot B \mid B \in SL_{m^2}(\mathbb{C})\}.$$

- It follows that $\widehat{\text{per}}_n$ lies in the closure of O.

Geometric Invariant Theory Approach

- Polynomial per $_n$ has far fewer automorphisms than det $_m$:
  - det $_m$ is invariant under the map $Y \mapsto CYD^{-1}$ where $\det C = \det D \neq 0$.
  - per $_n$ is invariant under the map $X \mapsto CYD^{-1}$ where both C and D are either diagonal or permutation matrices.
- For any point in O, its set of automorphisms is a conjugate of the set of automorphisms of det $_m$.

Geometric Invariant Theory Approach

- Polynomial per $_n$ has far fewer automorphisms than det $_m$:
  - det $_m$ is invariant under the map $Y \mapsto CYD^{-1}$ where $\det C = \det D \neq 0$.
  - per $_n$ is invariant under the map $X \mapsto CYD^{-1}$ where both C and D are either diagonal or permutation matrices.
- For any point in O, its set of automorphisms is a conjugate of the set of automorphisms of det $_m$.

Geometric Invariant Theory Approach

- Polynomial per_n has far fewer automorphisms than $\det_m$:
  - $\det_m$ is invariant under the map $Y \mapsto CYD^{-1}$ where $\det C = \det D \neq 0$.
  - per_n is invariant under the map $X \mapsto CYD^{-1}$ where both C and D are either diagonal or permutation matrices.
- For any point in O, its set of automorphisms is a conjugate of the set of automorphisms of $\det_m$.

Geometric Invariant Theory Approach

Polynomial $\widehat{\text{per}}_n$ has several additional automorphisms due to additional $m - n$ variables.

Hypothesis. For small m, a point that has the set of automorphisms of $\widehat{\text{per}}_n$ cannot occur in the closure of O.

Geometric Invariant Theory Approach

Polynomial $\widehat{\text{per}}_n$ has several additional automorphisms due to additional $m - n$ variables.

Hypothesis. For small m, a point that has the set of automorphisms of $\widehat{\text{per}}_n$ cannot occur in the closure of O.

Outline

1. Determinant and Permanent
2. Complexity Notions
3. Known Lower Bounds on Complexity of Permanent
4. Proving Strong Lower Bounds on Determinant Complexity
5. Proving Strong Lower Bounds on Circuit Complexity
6. Proving Hardness of Permanent Polynomial

Derandomization and Lower Bounds

- Kabanets and Impagliazzo (2003) showed a connection between derandomization of Identity Testing problem and lower bounds on arithmetic circuits:
  - If Identity Testing problem can be solved deterministically in polynomial time then NEXP has superpolynomial circuit complexity.
- This connection can be made stronger via black-box derandomization, or equivalently, pseudo-random generators.

Derandomization and Lower Bounds

- Kabanets and Impagliazzo (2003) showed a connection between derandomization of Identity Testing problem and lower bounds on arithmetic circuits:
  - If Identity Testing problem can be solved deterministically in polynomial time then NEXP has superpolynomial circuit complexity.
- This connection can be made stronger via black-box derandomization, or equivalently, pseudo-random generators.

Identity Testing

Definition

Given a polynomial computed by an arithmetic circuit over field F, test if the polynomial is identically zero.

Pseudo-Random Generators Against Arithmetic Circuits

- Let $\mathcal{A}_F$ be a class of arithmetic circuits over field F with $\mathcal{A}_F^s$ denoting the subclass of $\mathcal{A}_F$ of circuits of size s.
- Let $f : \mathbb{N} \mapsto (F[y])^*$ be a function such that $f(s) = (p_{s,1}(y), \ldots, p_{s,s}(y), q_s(y))$ for all s.

Definition

Function f is a pseudo-random generator against $\mathcal{A}_F$ if

- Each $p_{s,i}(y)$ and $q_s(y)$ is of degree $s^{O(1)}$.
- For any circuit $C \in \mathcal{A}_F^s$ with $n \leq s$ inputs:

$$C(x_1, \ldots, x_n) = 0 \text{ iff } C(p_{s,1}(y), \ldots, p_{s,n}(y)) = 0 \ (mod\ q_s(y)).$$

Pseudo-Random Generators Against Arithmetic Circuits

- Let $\mathcal{A}_F$ be a class of arithmetic circuits over field F with $\mathcal{A}_F^s$ denoting the subclass of $\mathcal{A}_F$ of circuits of size s.
- Let $f : \mathbb{N} \mapsto (F[y])^*$ be a function such that $f(s) = (p_{s,1}(y), \ldots, p_{s,s}(y), q_s(y))$ for all s.

Definition

Function f is a pseudo-random generator against $\mathcal{A}_F$ if

- Each $p_{s,i}(y)$ and $q_s(y)$ is of degree $s^{O(1)}$.
- For any circuit $C \in \mathcal{A}_F^s$ with $n \leq s$ inputs:

$$C(x_1, \ldots, x_n) = 0 \text{ iff } C(p_{s,1}(y), \ldots, p_{s,n}(y)) = 0 \ (mod\ q_s(y)).$$

Pseudo-Random Generators Against Arithmetic Circuits

- Let $\mathcal{A}_F$ be a class of arithmetic circuits over field F with $\mathcal{A}_F^s$ denoting the subclass of $\mathcal{A}_F$ of circuits of size s.
- Let $f : \mathbb{N} \mapsto (F[y])^*$ be a function such that $f(s) = (p_{s,1}(y), \ldots, p_{s,s}(y), q_s(y))$ for all s.

Definition

Function f is a pseudo-random generator against $\mathcal{A}_F$ if

- Each $p_{s,i}(y)$ and $q_s(y)$ is of degree $s^{O(1)}$.
- For any circuit $C \in \mathcal{A}_F^s$ with $n \leq s$ inputs:

$$C(x_1, \ldots, x_n) = 0 \text{ iff } C(p_{s,1}(y), \ldots, p_{s,n}(y)) = 0 \ (mod\ q_s(y)).$$

Existance of Pseudo-Random Generators

- Schwartz-Zippel provide an efficient randomized algorithm to test if a given circuit computes zero polynomial.
- The same argument shows that a random choice of f is a pseudo-random generator against the entire class of arithmetic circuits with good probability.

Existance of Pseudo-Random Generators

- Schwartz-Zippel provide an efficient randomized algorithm to test if a given circuit computes zero polynomial.
- The same argument shows that a random choice of f is a pseudo-random generator against the entire class of arithmetic circuits with good probability.

Efficiently Computable Pseudo-Random Generators

- A pseudo-random generator that can be quickly computed is very useful.

Definition

Function f is an efficiently computable pseudo-random generator against $\mathcal{A}_F$ if

- It is a pseudo-random generator against $\mathcal{A}_F$.
- $f(s)$ can be computed in time $s^{O(1)}$.

Efficiently Computable Pseudo-Random Generators

- A pseudo-random generator that can be quickly computed is very useful.

Definition

Function f is an efficiently computable pseudo-random generator against $\mathcal{A}_F$ if

- It is a pseudo-random generator against $\mathcal{A}_F$.
- $f(s)$ can be computed in time $s^{O(1)}$.

Efficiently Computable Pseudo-Random Generators

- If there exist efficiently computable pseudo-random generators against the entire class of arithmetic circuits then:
  - The identity testing problem can be solved in determinstic polynomial-time.
  - There exists a multilinear polynomial in EXP that cannot be computed by subexponential sized arithmetic circuits.

Efficiently Computable Pseudo-Random Generators

- If there exist efficiently computable pseudo-random generators against the entire class of arithmetic circuits then:
  - The identity testing problem can be solved in determinstic polynomial-time.
  - There exists a multilinear polynomial in EXP that cannot be computed by subexponential sized arithmetic circuits.

Efficiently Computable Pseudo-Random Generators

- If there exist efficiently computable pseudo-random generators against the entire class of arithmetic circuits then:
  - The identity testing problem can be solved in determinstic polynomial-time.
  - There exists a multilinear polynomial in EXP that cannot be computed by subexponential sized arithmetic circuits.

A Polynomial With High Circuit Complexity

- Let f be an efficiently computable pseudo-random generator against $\mathcal{A}_F$.
- Let the degree of all polynomials in $p_{s,1}(y)$, …, $p_{s,s}(y)$ be bounded by $d = s^{O(1)}$ and $m = \log d = O(\log s)$.
- Define polynomial r_{2m} as:

$$r_{2m}(x_1, x_2, \ldots, x_{2m}) = \sum_{S \subseteq [1,2m]} c_S \prod_{i \in S} x_i.$$

- Coefficients $c_S \in F$ satisfy:

$$\sum_{S \subseteq [1,2m]} c_S \prod_{i \in S} p_{s,i}(y) = 0.$$

A Polynomial With High Circuit Complexity

- Let f be an efficiently computable pseudo-random generator against $\mathcal{A}_F$.
- Let the degree of all polynomials in $p_{s,1}(y)$, ..., $p_{s,s}(y)$ be bounded by $d = s^{O(1)}$ and $m = \log d = O(\log s)$.
- Define polynomial r_{2m} as:

$$r_{2m}(x_1, x_2, \ldots, x_{2m}) = \sum_{S \subseteq [1,2m]} c_S \prod_{i \in S} x_i.$$

- Coefficients $c_S \in F$ satisfy:

$$\sum_{S \subseteq [1,2m]} c_S \prod_{i \in S} p_{s,i}(y) = 0.$$

A Polynomial With High Circuit Complexity

- Let f be an efficiently computable pseudo-random generator against $\mathcal{A}_F$.
- Let the degree of all polynomials in $p_{s,1}(y)$, ..., $p_{s,s}(y)$ be bounded by $d = s^{O(1)}$ and $m = \log d = O(\log s)$.
- Define polynomial r_{2m} as:

$$r_{2m}(x_1, x_2, \ldots, x_{2m}) = \sum_{S \subseteq [1,2m]} c_S \prod_{i \in S} x_i.$$

- Coefficients $c_S \in F$ satisfy:

$$\sum_{S \subseteq [1,2m]} c_S \prod_{i \in S} p_{s,i}(y) = 0.$$

A Polynomial With High Circuit Complexity

- Let f be an efficiently computable pseudo-random generator against $\mathcal{A}_F$.
- Let the degree of all polynomials in $p_{s,1}(y)$, ..., $p_{s,s}(y)$ be bounded by $d = s^{O(1)}$ and $m = \log d = O(\log s)$.
- Define polynomial r_{2m} as:

$$r_{2m}(x_1, x_2, \ldots, x_{2m}) = \sum_{S \subseteq [1,2m]} c_S \prod_{i \in S} x_i.$$

- Coefficients $c_S \in F$ satisfy:

$$\sum_{S \subseteq [1,2m]} c_S \prod_{i \in S} p_{s,i}(y) = 0.$$

A Polynomial With High Circuit Complexity

It can be shown that:

- A non-zero r_{2m} always exists.
- Polynomial r_{2m} can be computed by exponential size arithmetic circuits.
- Circuit complexity of r_{2m} is more than $s = 2^{O(m)}$.

A Polynomial With High Circuit Complexity

It can be shown that:

- A non-zero r_{2m} always exists.
- Polynomial r_{2m} can be computed by exponential size arithmetic circuits.
- Circuit complexity of r_{2m} is more than $s = 2^{O(m)}$.

A Polynomial With High Circuit Complexity

It can be shown that:

- A non-zero r_{2m} always exists.
- Polynomial r_{2m} can be computed by exponential size arithmetic circuits.
- Circuit complexity of r_{2m} is more than $s = 2^{O(m)}$.

Outline

1. Determinant and Permanent
2. Complexity Notions
3. Known Lower Bounds on Complexity of Permanent
4. Proving Strong Lower Bounds on Determinant Complexity
5. Proving Strong Lower Bounds on Circuit Complexity
6. Proving Hardness of Permanent Polynomial

A Simpler Task

Construct an efficiently computable pseudo-random generator against the class of size s, depth $\omega(1)$ arithmetic circuits of degree s.

This Yields Superpolynomial Lower Bounds

There exists an efficiently computable pseudo-random generator against the class of size s, depth $\omega(1)$ arithmetic circuits of degree s

$$\Downarrow$$

There is a multilinear polynomial r_{2m} of circuit complexity $2^{O(m)}$ that cannot be computed by size $2^{o(m)}$, depth $\omega(1)$ circuits

$$\Downarrow$$

Polynomial r_{2m} cannot be computed by any size $m^{O(1)}$ arithmetic circuit

This Yields Superpolynomial Lower Bounds

There exists an efficiently computable pseudo-random generator against the class of size s, depth $\omega(1)$ arithmetic circuits of degree s

$$\Downarrow$$

There is a multilinear polynomial r_{2m} of circuit complexity $2^{O(m)}$ that cannot be computed by size $2^{o(m)}$, depth $\omega(1)$ circuits

$$\Downarrow$$

Polynomial r_{2m} cannot be computed by any size $m^{O(1)}$ arithmetic circuit

This Yields Superpolynomial Lower Bounds

There exists an efficiently computable pseudo-random generator against the class of size s, depth $\omega(1)$ arithmetic circuits of degree s

$$\Downarrow$$

There is a multilinear polynomial r_{2m} of circuit complexity $2^{O(m)}$ that cannot be computed by size $2^{o(m)}$, depth $\omega(1)$ circuits

$$\Downarrow$$

Polynomial r_{2m} cannot be computed by any size $m^{O(1)}$ arithmetic circuit

Connecting to Permanent

- Can each r_{2m} be computed as permanent of a small matrix?
- Recall:

$$r_{2m}(x_1, x_2, \ldots, x_{2m}) = \sum_{S \subseteq [1,2m]} c_S \prod_{i \in S} x_i.$$

- Define

$$\hat{r}_{4m}(x_1, \ldots, x_{2m}, y_1, \ldots, y_{2m}) = c(y_1, \ldots, y_{2m}) \prod_{i=1}^{2m} (y_i x_i - y_i + 1),$$

where $c(b_1, \ldots, b_{2m}) = c_S$, $S = \{i \mid b_i = 1\}$.

- Then:

$$r_{2m}(x_1, x_2, \ldots, x_{2m}) = \sum_{y_1=0}^{1} \cdots \sum_{y_{2m}=0}^{1} \hat{r}_{4m}(x_1, \ldots, x_{2m}, y_1, \ldots, y_{2m}).$$

Connecting to Permanent

- Can each r_{2m} be computed as permanent of a small matrix?
- Recall:

$$r_{2m}(x_1, x_2, \ldots, x_{2m}) = \sum_{S \subseteq [1,2m]} c_S \prod_{i \in S} x_i.$$

- Define

$$\hat{r}_{4m}(x_1, \ldots, x_{2m}, y_1, \ldots, y_{2m}) = c(y_1, \ldots, y_{2m}) \prod_{i=1}^{2m} (y_i x_i - y_i + 1),$$

where $c(b_1, \ldots, b_{2m}) = c_S$, $S = \{i \mid b_i = 1\}$.

- Then:

$$r_{2m}(x_1, x_2, \ldots, x_{2m}) = \sum_{y_1=0}^{1} \cdots \sum_{y_{2m}=0}^{1} \hat{r}_{4m}(x_1, \ldots, x_{2m}, y_1, \ldots, y_{2m}).$$

Connecting to Permanent

- Can each r_{2m} be computed as permanent of a small matrix?
- Recall:

$$r_{2m}(x_1, x_2, \ldots, x_{2m}) = \sum_{S \subseteq [1,2m]} c_S \prod_{i \in S} x_i.$$

- Define

$$\hat{r}_{4m}(x_1, \ldots, x_{2m}, y_1, \ldots, y_{2m}) = c(y_1, \ldots, y_{2m}) \prod_{i=1}^{2m} (y_i x_i - y_i + 1),$$

where $c(b_1, \ldots, b_{2m}) = c_S$, $S = \{i \mid b_i = 1\}$.

- Then:

$$r_{2m}(x_1, x_2, \ldots, x_{2m}) = \sum_{y_1=0}^{1} \cdots \sum_{y_{2m}=0}^{1} \hat{r}_{4m}(x_1, \ldots, x_{2m}, y_1, \ldots, y_{2m}).$$

Connecting to Permanent

- Can each r_{2m} be computed as permanent of a small matrix?
- Recall:

$$r_{2m}(x_1, x_2, \ldots, x_{2m}) = \sum_{S \subseteq [1,2m]} c_S \prod_{i \in S} x_i.$$

- Define

$$\hat{r}_{4m}(x_1, \ldots, x_{2m}, y_1, \ldots, y_{2m}) = c(y_1, \ldots, y_{2m}) \prod_{i=1}^{2m} (y_i x_i - y_i + 1),$$

where $c(b_1, \ldots, b_{2m}) = c_S$, $S = \{i \mid b_i = 1\}$.

- Then:

$$r_{2m}(x_1, x_2, \ldots, x_{2m}) = \sum_{y_1=0}^{1} \cdots \sum_{y_{2m}=0}^{1} \hat{r}_{4m}(x_1, \ldots, x_{2m}, y_1, \ldots, y_{2m}).$$

Connecting to Permanent

- By Valiant (1979), if $\hat{r}_{4m}$ has circuit complexity $m^{O(1)}$ then r_{2m} can be computed as permanent of a matrix of size $m^{O(1)}$.
- So a pseudo-random generator such that $\hat{r}_{4m}$ has circuit complexity $m^{O(1)}$ implies that Permanent has circuit complexity $m^{\omega(1)}$.

Connecting to Permanent

- By Valiant (1979), if $\hat{r}_{4m}$ has circuit complexity $m^{O(1)}$ then r_{2m} can be computed as permanent of a matrix of size $m^{O(1)}$.
- So a pseudo-random generator such that $\hat{r}_{4m}$ has circuit complexity $m^{O(1)}$ implies that Permanent has circuit complexity $m^{\omega(1)}$.

Current Status: Small Depth Circuits

- We know efficiently computable pseudo-random generators against size s, depth two arithmetic circuits.
- Still some way to go!

Current Status: Small Depth Circuits

- We know efficiently computable pseudo-random generators against size s, depth two arithmetic circuits.
- Still some way to go!

Current Status: Large Depth but Restricted Class of Circuits

- A-Kayal-Saxena (2002) constructed an efficiently computable pseudo-random generator against a very special class of circuits.
- This contained circuits computing the polynomial $(1+x)^m - x^m - 1$ over ring Z_m.
- The pseudo-random generator is:

$$f(s) = (y, 0, \ldots, 0, q_s(y)), q_s(y) = y^{16s^5} \prod_{t=1}^{16s^5} \prod_{a=1}^{4s^4} ((y-a)^t - 1).$$

- This derandomized a primality testing algorithm.

Current Status: Large Depth but Restricted Class of Circuits

- A-Kayal-Saxena (2002) constructed an efficiently computable pseudo-random generator against a very special class of circuits.
- This contained circuits computing the polynomial $(1+x)^m - x^m - 1$ over ring Z_m.
- The pseudo-random generator is:

$$f(s) = (y, 0, \ldots, 0, q_s(y)), q_s(y) = y^{16s^5} \prod_{t=1}^{16s^5} \prod_{a=1}^{4s^4} ((y-a)^t - 1).$$

- This derandomized a primality testing algorithm.

A Conjecture

Define

$$f(s, k) = (y, y^k, y^{k^2}, \ldots, y^{k^{s-1}}, y^r - 1),$$

where $r \geq s^4$ is a prime and $1 \leq k < r$.

Conjecture

Function f is a pseudo-random generator against arithmetic circuits of size s, depth $\omega(1)$, and degree s.

A Conjecture

Define

$$f(s,k) = (y, y^k, y^{k^2}, \ldots, y^{k^{s-1}}, y^r - 1),$$

where $r \geq s^4$ is a prime and $1 \leq k < r$.

Conjecture

Function f is a pseudo-random generator against arithmetic circuits of size s, depth $\omega(1)$, and degree s.