Complexity Theory of Polynomial-Time Problems

Lecture 3: The polynomial method  
Part I: Orthogonal Vectors

Sebastian Krinninger
Organization of lecture

• No lecture on 26.05. (State holiday)
• 2nd exercise sheet: Next week
• Tutorials:
  • New slot: Friday, 12:15 - 14:00, U12 E1.1, biweekly
  • Fr, 13.05. (tomorrow)
  • Fr, 03.06.
  • Etc.
The polynomial method

• Recently developed technique in Algorithm Design
• Current fastest algorithms for
  • All-Pairs Shortest Paths [Williams 14]
  • Orthogonal Vectors [Abboud/Williams/Yu 15]
  • Hamming Nearest Neighbors [Alman/Williams 15]
• Two main tools
  1. Razborov-Smolensky from Circuit Complexity
  2. Fast rectangular matrix multiplication
Reminder: Orthogonal Vectors Problem

Input: Two sets $A, B \subseteq \{0,1\}^d$ of $d$-dimensional 0/1-vectors of size $n$

Output: Is there a pair $a \in A, b \in B$ s.t. $a$ and $b$ are orthogonal?

$\exists a \in A, b \in B: \langle a, b \rangle = 0$

$\sum_{k=1}^{d} a[k] \cdot b[k] = 0$

$\forall 1 \leq k \leq d: (a[k] = 0) \lor (b[k] = 0)$

Trivial algorithms:

• $O(n^2 d)$

• $O(2^d n)$

Interesting Regime: $d = c \log n$
Today’s result

Reminder:

**Conjecture**: There is no algorithm for the orthogonal vectors problem with running time $O(n^{2-\epsilon}\text{poly}(d))$ for any $\epsilon > 0$.

State of the art:

**Theorem**: There is an algorithm for the orthogonal vectors problem with running time $n^{2-1/O(\log(d/\log n))}$.

**In this lecture**: $n^{2-1/O(\log d)}$

Algorithm is randomized and correct with high probability, i.e., probability $\geq 1 - 1/n$
Overview

1. Reduce problem to many subproblems of very small size
2. Precompute small circuits for solving subproblems
3. Evaluate circuits with probabilistic polynomials of low degree
4. Evaluate polynomials using fast rectangular matrix multiplication
Overview

1. Reduce problem to many subproblems of very small size
2. Precompute small circuits for solving subproblems
3. Evaluate circuits with probabilistic polynomials of low degree
4. Evaluate polynomials using fast rectangular matrix multiplication
Dividing into smaller subproblems

1. Divide $A$ and $B$ into $q = \lceil \frac{n}{s} \rceil$ subsets of size $\leq s$: 
   
   $A_1, ..., A_q$ and $B_1, ..., B_q$

2. Construct a polynomial $P(a_1[1], ... a_1[d], ..., a_s[1], ..., a_s[d],$
   
   $b_1[1], ... b_1[d], ..., b_s[1], ..., b_s[1])$

   $P(A_i, B_j) = 1$ if and only if $A_i, B_j$ contains orthogonal pair 

   ...only with high probability

3. For every pair of subsets $A_i, B_j$: evaluate $P$ on $A_i, B_j$

   ...simultaneously! $\rightarrow O(\frac{n^2}{s^2} \text{ polylog}(n))$

4. Return “yes” if some $A_i, B_j$ contains orthogonal pair, “no” otherwise

   We set $s = 2^\epsilon \frac{\log n}{\log d} = n^\epsilon / \log d$ for some sufficiently small $\epsilon$
Questions

1. How to construct suitable polynomial $P$?
2. How to evaluate $P$ fast on all pairs of inputs?
Overview

1. Reduce problem to many subproblems of very small size
2. Precompute small circuits for solving subproblems
3. Evaluate circuits with probabilistic polynomials of low degree
4. Evaluate polynomials using fast rectangular matrix multiplication
Boolean circuits

Boolean circuit
• Directed acyclic graph
• Sources: input bits
• Sink: output bit
• Inner nodes: Boolean operations
• AND: $\wedge$
• OR: $\lor$
• Arbitrary “fan-in”
Circuit for checking orthogonality of vectors

Are $x$ and $y$ orthogonal?

$x$ and $y$ orthogonal iff

$\neg \exists i: x[i] = 1 \land y[i] = 1$

Output bit $z = 1$ iff

$x$ and $y$ orthogonal

AND of ORs with

- $2d$ negated inputs
- 1 output
Circuit for finding orthogonal pair

Is there an orthogonal pair?

Check orthogonality for every pair

\( x_1, y_1 \) orthogonal OR
\( x_1, y_2 \) orthogonal OR
... OR
\( x_s, y_s \) orthogonal?

OR of ANDs of ORs

\( 2ds^2 \) negated inputs
Overview

1. Reduce problem to many subproblems of very small size
2. Precompute small circuits for solving subproblems
3. Evaluate circuits with probabilistic polynomials of low degree
4. Evaluate polynomials using fast rectangular matrix multiplication
From circuits to polynomials

• Obtain polynomial over $F_2$ outputting 1 if and only if circuit outputs 1
• $F_2$: Field of $\{0, 1\}$ with operations $\oplus$ and $\cdot$
• $\oplus$ is XOR-operation:
  • $0 \oplus 0 = 0, 1 \oplus 0 = 1, 0 \oplus 1 = 1, 1 \oplus 1 = 0$
  • XOR of multiple variables:
    $x_1 \oplus x_2 \oplus \cdots \oplus x_k = 1$ if and only if odd number of $x_i$’s is 1
• Expanded polynomials:
  • $a \cdot (b \oplus c) \cdot (a \oplus b \oplus d) = ac \oplus abc \oplus abd \oplus acd$
  • XOR of monomials
  • Goal: Few monomials allows fast evaluation later
Representing circuit by polynomial

Straightforward approach:

• AND: \( a \land b \Rightarrow a \cdot b \)

• Negation: \( \neg a \Rightarrow 1 \oplus a \) \hspace{1cm} \text{(addition=subtraction in } F_2 \text{)}

• OR: \( a \lor b \Rightarrow \neg(\neg a \land \neg b) \) \hspace{1cm} \text{(DeMorgan’s Law)}

Example: Bottom-level of circuit

\( \neg a \lor \neg b \Rightarrow 1 \oplus a \cdot b \)
Expanding a multiplication (distributive law)

\[(x_{1,1} \oplus \cdots \oplus x_{1,k}) \cdot (x_{2,1} \oplus \cdots \oplus x_{2,k}) \cdot \cdots \cdot (x_{t,1} \oplus \cdots \oplus x_{t,k}) = x_{1,1} \cdot x_{2,1} \cdot \cdots \cdot x_{t,1} \oplus \cdots \oplus x_{1,k} \cdot x_{2,k} \cdot \cdots \cdot x_{t,k}\]

- **AND of** \(t\) **XORs**: Each XOR has \(k\) inputs
- **XOR of** \(k^t\) **ANDs**: Each AND has \(t\) inputs

Running time: \(O(k^t \cdot \text{deg})\) where \(\text{deg}\) is degree after expansion (maximum size of any monomial); here \(\text{deg} \leq t\)
Razborov/Smolensky trick [Raz87] [Smo87]

Naïve representation of OR

DeMorgan:

\[ OR(x_1, \ldots, x_k) = 1 \oplus \prod_{i=1}^{k} (1 \oplus x_i) \]

After expansion: \(2^k\) monomials

Probabilistic representation of OR

Parameter \(t\)

Fewer monomials, correct whp

OR of \(t\) XORs

Add each edge

With probability \(\frac{1}{2}\)
Correct representation with high probability

Case 1: $x_1 \lor \cdots \lor x_k = 0$

Easy case: each XOR outputs 0, top OR outputs 0

Case 2: $x_1 \lor \cdots \lor x_k = 1$

Let $X$ be set of inputs with $x_i = 1$

For each XOR:
- If XOR has odd number of links to $X$: XOR outputs 1 (good event: top OR outputs 1)
- If XOR has even number of links to $X$: XOR outputs 0 (bad event!)

Probability that XOR has even number of links to $X$:
$= 1/2$ because last element of $X$ “decides” whether number of links is even or odd (each with prob. 1/2)

$\Rightarrow$ Probability that all XORs output 0: $= \left(\frac{1}{2}\right)^t = \frac{1}{2^t}$

$\Rightarrow$ Probability that OR outputs 1: $= 1 - \frac{1}{2^t}$
Bounding number of monomials

Formal definition of polynomial
For $i = 1 \ldots t, j = 1 \ldots k$:
- With probability $\frac{1}{2}$: Set $r_{i,j} = 1$
- Otherwise: Set $r_{i,j} = 0$

Polynomial: $OR_t(x_1, \ldots, x_k) =$
$$1 \oplus \prod_{i=1}^{k} (1 \oplus \bigoplus_{j=1}^{k} r_{i,j} \cdot x_i)$$

After expansion: $(k + 1)^t$ monomials
Formal definition

**Definition:** Let $C$ be a Boolean circuit with $k$ input gates and let $D$ be a finite distribution of polynomials on $k$ variables over a ring $R$ containing 0 and 1\(^{(\ast)}\). The distribution $D$ is a probabilistic polynomial over $R$ representing $C$ with error $\delta$ if for all $(x_1, \ldots, x_k) \in \{0,1\}^k$:

$$\Pr_{p \sim D}[p(x_1, \ldots, x_k) = C(x_1, \ldots, x_k)] > 1 - \delta.$$

Example: OR-gate represented by

$$OR_t(x_1, \ldots, x_k) = 1 \oplus \prod_{i=1}^{t}(1 \oplus \bigoplus_{j=1}^{k} r_{i,j} \cdot x_i) \text{ with error } \delta = 1 - \frac{1}{2^t}$$

\(^{(\ast)}\) In our case, $R$ is the field $F_2$
Representing OV circuit I

Are $x$ and $y$ orthogonal?

Bottom OR:
$$\neg x[k] \lor \neg y[k] \Rightarrow 1 \oplus x[k] \cdot y[k]$$

Middle AND:
1. DeMorgan
2. Razborov/Smolensky with $t_1 = 3 \log s$

Number of monomials:
$$(d + 1)^{t_1}$$
Representing OV circuit II

Is there an orthogonal pair?

Middle ANDs:
XOR of \(s^2\) polynomials, each with \((d + 1)^{t_1}\) monomials
\[\Rightarrow s^2 (d + 1)^{t_1}\] monomials

Top OR:
Raz/Smol with \(t_2 = 2\)
\[\Rightarrow (s^2(d + 1)^{t_1})^{t_2}\] monomials
\[= s^4(d + 1)^6 \log s\]
Analysis of error

We apply Razborov/Smolensky

- $s^2$ times with $t_1 = 3 \log s$
- 1 time with $t_2 = 2$

**Union Bound:** $\Pr(X \cup Y) \leq \Pr(X) + \Pr(Y)$

Probability of error: $\leq \frac{s^2}{2t_1} + \frac{1}{2t_2} = \frac{s^2}{s^3} + \frac{1}{4} = \frac{1}{s} + \frac{1}{4} \leq \frac{1}{3}$

($1/s$ is small enough for instances with sufficiently large $n$)
Plugging in the right values

\[ s = 2^\varepsilon \log n / \log d \]
\[ \varepsilon = 1/160 \]

#monomials: \[ m \leq s^4 (d + 1)^6 \log s \leq s^4 (d + 1)^6 \varepsilon \log n / \log d \]

\[ \leq 4\varepsilon \log n + 12\varepsilon \log n = 0.1 \log n \]

\[ \Rightarrow m \leq n^{0.1} \]
Running time for expanding polynomial

We explicitly have to expand our polynomial into XOR of monomials!
Running time dominated by applications of distributive law

1st expansion (repeated $s^2$ times):
- Degree after expansion: $O(t_1)$
- Total time: $O(s^2(d + 1)t_1t_1^2)$

2nd expansion:
- Degree after expansion: $O(t_1t_2)$
- Running time: $O \left( (s^2(d + 1)t_1^2 t_1^2) \right) \leq O(n^{0.1} t_1^2 t_2^2) \leq O(n^{0.1} \log^2 n)$

$\Rightarrow$ Total time: $O(n^{0.1} \log^2 n)$ (negligible)
Summary for probabilistic polynomial

We can construct polynomial $P$ over $F_2$ with $2sd$ inputs such that, given two sets $A', B' \subseteq \{0,1\}^d$ of $d$-dimensional 0/1-vectors of size $s$, with probability $> \frac{2}{3}$: $P(A', B') = 1$ iff $A'$ and $B'$ have orthogonal pair.
Overview

1. Reduce problem to many subproblems of very small size
2. Precompute small *circuits* for solving subproblems
3. Evaluate circuits with *probabilistic polynomials of low degree*
4. Evaluate polynomials using fast *rectangular* matrix multiplication
Fast matrix multiplication

• Goal: Compute $C = A \times B$ where $A$ and $B$ are $n \times n$ matrices
• Naïve algorithm: $O(n^3)$
• Strassen’s algorithm: $O(n^{2.807})$
• Current fastest: $O(n^{2.373})$
• Best we can hope for: $O(n^2)$
Lemma: There is an algorithm for multiplying an $N \times N^{0.17}$ matrix with an $N^{0.17} \times N$ matrix in time $O(N^2 \log^2 n)$.

Also works for finite fields such as $F_2$!
Fast evaluation of polynomial

- Given: Polynomial $P(x[1], ..., x[K], y[1], ..., y[K])$ over $F_2$
- With at most $N^{0.1}$ monomials
- Two sets of inputs:
  - $X = \{x_1, ..., x_N\} \subseteq \{0,1\}^K$, $Y = \{y_1, ..., y_N\} \subseteq \{0,1\}^K$
- Evaluate $P$ on all pairs $(x_i, y_j) \in X \times Y$ simultaneously
  in time $O(N^2 \text{polylog}(n))$
Reduction to matrix multiplication

\[ x_1 \ldots x_N \times y_1 \ldots y_N \]

Entry \((i, j)\): Restriction of \(i\)-th monomial to input \(y_j\)

Entry \((i, j)\): Restriction of \(j\)-th monomial to input \(x_i\)
Evaluating OV-polynomial on all subgroups

1. Divide $A$ and $B$ into $q = \lceil \frac{n}{s} \rceil$ subsets of size $\leq s$:
   $A_1, ..., A_q$ and $B_1, ..., B_q$

2. Construct a polynomial $P(a_1[1], ..., a_1[d], ..., a_q[1] ..., a_q[d],$
   $b_1[1], ..., b_1[d], ..., b_q[1] ..., b_q[1])$
   $P(A_i, B_j) = 1$ if and only if $A_i, B_j$ contains orthogonal pair

3. For every pair of subsets $A_i, B_j$: evaluate $P$ on $A_i, B_j$
   $P$ has $\leq n^{0.1}$ monomials
   Simultaneous evaluation in time $O(n^2/s^2 \text{ polylog}(n)) \leq n^{2-1/O(\log d)}$

   $s = 2^\epsilon \log n / \log d = n^{\epsilon/\log d}$ for $\epsilon = 1/160$
Remarks

Correctness with high probability
- Polynomial is only correct with probability $\geq 2/3$
- Amplify the success probability by repeating with $10 \log n$ independent polynomials and taking majority value
- $\implies$ Chernoff Bound

Faster algorithm:
- $n^{2-1/\Theta(\log(d/\log n))}$
- Just needs better estimate for number of monomials and slightly different choice of $s$