A introduction to:
Algebrization: A New Barrier in Complexity Theory[1]

Yichen Nie

June 22, 2024

Contents

All section before **6 The Integers Case** contains 29 pages, they cover algebrization considering A's extension on finite fields and include 11 out of 15 important results. The essay will focus on the pages and provide an introduction to their contents.

1 Abstract

There are two barriers in complexity theory: *relativization* and *natural proofs*. Over the last decade, some result in complexity theory have overcome the two barriers simultaneously. In Aaronson and Wigderson's paper, they give a third barrier: *Algebrization*. The oracle is not both A like in *relativization*, in *Algebrization*, one complexity class is give A and another is give $\tilde{A}$ a low-degree extension of A over a finite field or ring.

In the paper, they first show result based on arithmetization do algebrize, and then show major open problem including P versus NP require *non-algebrizing technique*. Furthermore, the paper give lower bound of *algebraic query complexity* and reveal its connection to communication complexity.

2 Introduction

2.1 The history of barrier

In the history of the P versus NP problem, after numerous attempts to prove it, barriers—in Aaronson and Wigderson's words "meta-discoveries"—emerged:

A The *relativization* barrier

In Baker,Gill and Solovay's paper[2], they showed that method like diagonalization cannot use to solve the P versus NP problem. The *relativization* barrier consider a relative world, in the theory of complexity, the world is the oracle. With different oracle, a Turning machine can have different power in one step's computation. A relativizing technique will prove the relationship between two complexity class A and B for all relative world. So if one shows A,B's relation can't be the same for all relative world, then non-relativizing technique must be used.

B the *natural proofs* barrier

In 1997, Razborov and Rudich[3] give a new barrier for circuit lower bound. That is if we wish to prove a property P for boolean function does not exists in a class of boolean circuits, the property P cannot be natural:

- P(f) cannot be easily verifies
- a function f has property P is like a random function.

or it will contradict with the existence of strong one-way function.

2.2 The need for new barrier

In the early 1990's, researchers managed to prove IP = PSPACE[4, 5]. They use a technique called *arithmetization*, the key idea is: get information of φ from $\tilde{\varphi}$, its low-degree extension. This technique is non-relativizing. To avoid natural proof barrier, we can use diagonalization, since in diagonalization of $L \notin A$ L simulation all machines in A, so it can't looks like a random function. So combining *arithmetization* and *diagonalization* we get a technique avoiding both barriers. By giving a new barrier to this method, we can know how far the method can go.

2.3 Result of the Algebrization barrier

2.3.1 rough definition of algebrize

A is a oracle, $\tilde{A}$ is the low-degree extension of A.

- $C \subset D$ algebrizes if for all $A, \tilde{A}$ $C^A \subset D^{\tilde{A}}$
- $C \not\subset D$ algebrizes if for all $A, \tilde{A}$ $C^{\tilde{A}} \not\subset D^A$

Notice that the definition is asymmetric. Intuitively, we definition this way because a arithmetization technique will proof All $PSPACE^A \subset IP^{\tilde{A}}$. More detail of the definition will be discuss in **3 Preliminary**

2.3.2 result

- **result that Algebrizes:**
  - $PSPACE \subset IP$
  - $NEXP \subset MIP$
  - $MA_{EXP} \not\subset P/poly$
  - $PromiseMA \not\subset SIZE(n^k)$
- result require non-algebrizing techniques
  - $NP \not\subset P$
  - $NP \subset P$
  - $NP \subset BPP$
  - $P^{NP} \subset PP$
  - $NEXP \not\subset P/poly$
  - $NP \not\subset SIZE(n)$

  Aaronson and Wigderson summarized in the paper: *Algebrization provides nearly the precise limit on the non-relativizing techniques of the last two decades*

2.4 Techniques

2.4.1 Proving That Existing Results Algebrize

Interactive proof can be algebrized simply by change the normal boolean formula ψ to a boolean formula ψ^A that can use A gate(A is a oracle). Then we extend ψ^A and get $\tilde{\psi^A}$. And now use the LFKN protocol to solve it. Recall LFKN protocol, we only need to compute value of $\tilde{\psi^A}$ in the last step, and the extension fo A gate is actually $\tilde{A}$. Thus using this method we can proof many algebrize result.

And for separation results, we can find they contain a interactive part, thus can be solved.

2.4.2 Proving The Necessity of Non-Algebrizing Techniques

- 1 $C \subset D$ algebrize $\rightarrow$ for some $A, \tilde{A}$ $C^A \not\subset D^{\tilde{A}}$
- 2 $C \not\subset D$ algebrize $\rightarrow$ for some $A, \tilde{A}$ $C^{\tilde{A}} \subset D^A$

for 1 need to proof *algebraic oracle separations*, for 2 need to proof *algebraic oracle inclusions.*

For $NP \not\subset P$ does not algebrize, we consider A is a PSPACE-complete language, then $NP^{\tilde{A}} = P^A = PSPACE$.

For *algebraic oracle separations*, it is like *oracle separations*: $NP^B \not\subset P^B$. We proof it in two steps.

1 proof the lower bound on query complexity of some function

2 use the query complexity lower bound to diagonalize against a class of Turing machines.

this lead to the topic *algebraic query complexity*, the paper present two technique for lower-bounding algebraic query complexity

- constructive

  For query $y_1, \cdots, y_t \in \mathbb{F}^n$. we construct a low-degree polynomial p. That has $p(y_i) = 0, i \in [t]$ and also remains $2^n - t$ flexible boolean points. Thus the real polynomial will remain unknown on these points.

  this method is hard on random cases and integer cases.

- non-constructive

  It uses the connection between algebraic query complexity and communication complexity.

  If some property of A can be determined using T queries to a multilinear extension $\tilde{A}$ of A over the finite field F, then it can also be determined by Alice and Bob using O (Tnlog|F|) bits of communication.

  then the algebraic query complexity is lower-bounded by communication complexity lower bound.

  Remark. In **4 main content of the paper**, we will show that this result can be extend to random and quantum cases.(the M quries $\tilde{A}$ and Alice Bob are both random or quantum).

  In the non-constructive method, we get multilinear extensions and proof separation more easily, but the construction of separation language and oracle A is hard.

2.5 Related Work

Juma et al.[6] made the observation that, if the extension Ae is multilinear rather than multiquadratic, then oracle access to $\tilde{A}$ sometimes switches from being useless to being extraordinarily powerful.

3 Preliminary

This section will introduce formal definition of algebraic oracles, and various subtleties of the model.

Given a multivariate polynomial p (x1, . . . , xn), we define the multidegree of p, or mdeg (p), to be the maximum degree of any xi. We say p is multilinear if mdeg (p) ≤ 1, and multiquadratic if mdeg (p)≤ 2. Also, we call p an extension polynomial if $p(x) \in \{0, 1\}$ whenever $x \in \{0, 1\}^n$.

Definition 3.1 (Oracle). An oracle A is a collection of Boolean functions $A_m : \{0, 1\}^m \to \{0, 1\}$ one for each $m \in \mathbb{N}$. Then given a complexity class C, by C^A we mean the class of languages decidable by a C machine that can query A_m for any m of its choice. By $C^A[poly]$ we mean the class of languages decidable by a C machine that, on inputs of length n, can query A_m for any $m = O(poly(n))$. For classes C such that all computation paths are polynomially bounded (for example, P, NP, BPP, #P...), it is obvious that $C^A = C^{A[poly]}$

Definition 3.2 (Extension Oracle Over A Finite Field). Let $A_m : \{0, 1\}^m \to \{0, 1\}$ be a Boolean function, and let $\mathbb{F}$ be a finite field. Then an extension of A_m over $\mathbb{F}$ is a polynomial $\tilde{A}_{m,\mathbb{F}}\mathbb{F}^m \to \mathbb{F}$ such that $\tilde{A}_{m,\mathbb{F}}(x) = A_m(x)$ whenever $x \in \{0, 1\}^m$. Also, given an oracle $A = (A_m)$, an extension $\tilde{A}$ of A is a collection of polynomials $\tilde{A}_{m,\mathbb{F}} : F^m \to F$, one for each positive integer m and finite field $\mathbb{F}$, such that

(i) $\tilde{A}_{m,\mathbb{F}}$ is an extension of A_m for all m, $\mathbb{F}$, and

(ii) there exists a constant c such that $mdeg(\tilde{A}_{m,\mathbb{F}}) \le c$ for all m, $\mathbb{F}$.

Then given a complexity class C, by $C^{\tilde{A}}$ we mean the class of languages decidable by a C machine that can query $\tilde{A}_{m,\mathbb{F}}$ for any integer m and finite field $\mathbb{F}$. By $C^{\tilde{A}[poly]}$ we mean the class of languages decidable by a C machine that, on inputs of length n, can query $\tilde{A}_{m,\mathbb{F}}$ for any integer $m = O(poly(n))$ and finite field with $|\mathbb{F}| = 2^{O(m)}$.

notice that in $C^{\tilde{A}[poly]}$, the bit uses to query is restricted to poly(n).

mdeg($\tilde{A}$) denote the maximum multidegree of any $\tilde{A_m}$.

Definition 3.3 (Algebrization)**.** $C \subset D$ algebrize if $C^A \subset D^{\tilde{A}}$for all $A, \tilde{A}$. $C \not\subset D$ algebrize if $C^{\tilde{A}} \not\subset D^A$for all $A, \tilde{A}$.

The definition is asymmetric because in symmetric case, we still cannot show the result of interactive algebrize.

For any complexity class C,D, we call $C \subset D$ or $C \not\subset D$ a result. We define strictly algebrize to be like definition 3.3, but both oracle of C and D is $\tilde{A}$. Now we consider there sets

- R: contain all result that are relativize.
- SA: contain all result that are strictly algebrize.
- A: contain all result that algebrize.

We have

$$R \subset SA \subset A$$

So A can be seen as a extension of relativize. Also, A has weaker restriction than SA. So the all the non-algebrizing problem the paper proved will still be non-algebrizing if we change a definition.

4 main content of the paper

In this part, I will introduce its main idea and intuition of major result.

4.1 Why Existing Techniques Algebrize

4.1.1 Self-Correction for #P: Algebrizing

First, the paper define a a convenient #P-complete problem

Definition 4.1 (#FSAT)**.** An FSAT formula over the finite field $\mathbb{F}$, in the variables $x_1, \cdots, x_N$, is a circuit with unbounded fan-in and fan-out 1, where every gate is labeled with either + or ×, and every leaf is labeled with either an xi or a constant $c \in \mathbb{F}$. Such a formula represents a polynomial $p : \mathbb{F}^N \to \mathbb{F}$ in the obvious way. The size of the formula is the number of gates.

Now let $\#FSAT_{L,\mathbb{F}}$ be the following problem: given a polynomial $p : \mathbb{F}^N \to \mathbb{F}$ specified by an FSAT formula of size at most L, evaluate the sum

$$S(p) := \sum_{x_1,\cdots,x_N \in \{0,1\}} p(x_1, \cdots, x_N)$$

Also, let #FSAT be the same problem but where the input has the form $< L, \mathbb{F}, p >$ (i.e., L and F are given as part of the input). For the purpose of measuring time complexity, the size of an #FSAT instance is defined to be n := Llog |F|.

Observe that if p is represented by an F SAT formula of size L, then $deg(p) \leq L$. It is clear that #FSAT is #P-complete. Furthermore, Lund, Fortnow, Karloff, and Nisan[4] showed that #F SAT is self-correctible, in the following sense:

Theorem 4.1 ([4])**.** *There exists a polynomial-time randomized algorithm that,given any $\#FSAT_{L,\mathbb{F}}$ instance p with $char(\mathbb{F}) \geq 3L^2$ and any circuit C:*

(i) Outputs S (p) with certainty if C computes $\#FSAT_{L,\mathbb{F}}$

(ii) Outputs either S (p) or "FAIL" with probability at least 2/3, regardless of C

In theorem 4.1, C act as a Prover, try to proof the value of S(p). And the random algorithm can't be fooled about the value of S(p) with high probability.

The $FSAT^{\tilde{A}}$ is define as FSAT but can use $\tilde{A}$ gate: with arbitriy fan-in and 1 fan-out. $\#FSAT^{\tilde{A}}$ is define as $\#FSAT$, thus $\#FSAT^{\tilde{A}} \in \#P^{\tilde{A}}$

Proposition 4.1.1. *$\#FSAT^{\tilde{A}}$ is $\#P^{A}$-hard under randomized reductions.*

The paper give a $\#P^A$-complete problem: C^A boolean circuit over $x_1, \cdots, x_N$ and has oracle acess to A. Then computing $\sum_{x_1,\cdots,x_N\in\{0,1\}} C^A(x_1,\cdots,x_N)$ is a $\#P^A$-complete problem. We can arithmetization to reduce C^A to $FSAT^{\tilde{A}}$ formula. Thus the problem can be reduce to a $\#FSAT^{\tilde{A}}$. At last, use random algorithm to get $|\mathbb{F}|$ large enough to extend A

Theorem 4.2 (algebrizing version of Theorem 4.1)**.** *There exists a $BPP^{\tilde{A}}$ algorithm that, given any $\#FSAT^{\tilde{A}}_{L,\mathbb{F}}$ instance p with $char(F) \geq 3L^3 mdeg(\tilde{A})$ and any circuit $C^{\tilde{A}}$:*

(i) Outputs S (p) with certainty if $C^{\tilde{A}}$ computes $\#FSAT^{\tilde{A}}_{L,\mathbb{F}}$

(ii) Outputs either S (p) or "FAIL" with probability at least 2/3, regardless of $C^{\tilde{A}}$

The proof is basically identical to the usual proof of Lund et al.[4]. We use the $C^{\tilde{A}}$ as a prover, and run a protocol like LFKN protocol, not check the sum but calculate the sum, we win when we are not fooled by $C^{\tilde{A}}$ (calculate correctly or find out prover cheats). Each round we queries the circuit for the new polynomial, and in final case we check the values ourselves, because we have $\tilde{A}$, so we have a large probability to win.

Using this result and considering another language M.$< L, \mathbb{F}, p, k > \in M$ iff. $S(p) \geq k$. The paper proof a result

Theorem 4.3. *For all A,$\tilde{A}$, if $PP^{\tilde{A}} \subset P^{\tilde{A}}/poly$ then $P^{\#P^A} \subset MA^{\tilde{A}}$*

4.1.2 IP = PSPACE: Algebrizing

Theorem 4.4. *For all A,$\tilde{A}$,$P^{\#P^A} \subset IP^{\tilde{A}}$*

This is because in the proof of theorem 4.2, we actually give a interactive protocol of $\#FSAT$ using a $BPP^{\tilde{A}}$ verifer.

Theorem 4.5. *For all A,$\tilde{A}$, $PSPACE^{A[poly]} \subset IP^{\tilde{A}}$*

The proof is very similar to the original one. The verifier uses $\tilde{A}$ in the last step of protocol.

4.1.3 MIP = NEXP: Algebrizing

Theorem 4.6. *For all A,$\tilde{A}$, $NEXP^{A[poly]} \subset MIP^{\tilde{A}}$*

this is a algebrized version of Babai et al.'s result[7] In this paper, it is proved in several steps.

The first step is to define a convenient NEXP-complete problem

Definition 4.2 (hSAT)**.** Let an h-formula over the variables $x1, \cdots, xn \in \{0,1\}$ be a Boolean formula consisting of AND, OR, and NOT gates, as well as gates of fan-in n that compute a Boolean function $h : \{0,1\}^n \to \{0,1\}$ Then given an h-formula C^h,let hSAT be the problem of deciding whether there exists a Boolean function h,such that $C^h(x) = 0$ for all $x \in \{0,1\}$.

Babai et al. showed the following:

Lemma 4.7 ([7])**.** *hSAT is NEXP-complete.*

In this lemma h is function encode the nondeterministic guess and entire tableau of the computation with that guess. Define $hSAT^A$ to be hSAT that has $C^{h,A}$. So $hSAT^A$ is $NEXP^{A[poly]}$-complete.
Next step in Babai et al.'s proof is LFKN protocol with $\tilde{h}$ can verify whether $C^h(x) = 0$ for all x.

Lemma 4.8 ([7])**.** *Let $\tilde{h}$ be any low-degree extension of a Boolean function h. Then it is possible to verify, in $IP^{\tilde{h}}$ that $C^h(x) = 0$ for all x.*

We can arithmetize C^h, and use LFKN protocol to solve it. And because if C^h has other oracle gate A, $IP^{\tilde{A},\tilde{h}}$ can still use LFKN protocol to solve it, thus this result algebrizes.

Lemma 4.9 ([7])**.** *There exists a BPP^B algorithm that, given any oracle B input y.*

(i) Outputs B(p) if B is a low-degree polynomial.

(ii) Outputs "FAIL" with probability $\Omega(1/poly(n))$ if B differs from all low-degree polynomials on a $\Omega(1/poly(n))$ fraction of points.

combine lemma 4.8 and lemma 4.9 we get a MIP proof for $hSAT$. Actually it is a one proof P one oracle B interactive proof. The verifier runs a low-degree check(the algorithm in lemma 4.9), check if oracle B is like a low-degree. If not, reject. If is then B must be close to some $\tilde{h}$, we uses B to run LFKN protocol with P to verify $C^h(x) = 0$. In Babai et al.'s paper[7], they proof than the result of verify result after run LFKN protocol using B is close to the one using $\tilde{h}$. Notice that the low-degree is used only to check $\tilde{h}$, so the proof can be algebrize.

4.1.4 Recent Circuit Lower Bounds: Algebrizing

In this section, the paper gives several result using other algebrizing result and some simple derivation

Theorem 4.10. *For all $A, \tilde{A}$, constants k, $PP^{\tilde{A}} \not\subset SIZE^A(n^k)$*

Theorem 4.11. *For all $A, \tilde{A}$, $MA^{\tilde{A}}_{EXP} \not\subset P^A/poly$*

Theorem 4.12. *For all $A, \tilde{A}$, constants k, $PromiseMA^{\tilde{A}} \not\subset SIZE^A(n^k)$*

4.2 Lower Bounds on Algebraic Query Complexity

Definition 4.3 (Algebraic Query Complexity Over Fields)**.** Let $f : \{0,1\}^N \to \{0,1\}$be a Boolean function, let F be any field, and let c be a positive integer. Also, let M be the set of deterministic algorithms M such that $M^{\tilde{A}}$ outputs f (A) for every oracle A : : $\{0,1\}^n \to \{0,1\}$ and every finite field extension $\tilde{A} : \mathbb{F}^n \to \mathbb{F}$ of A with $mdeg(\tilde{A}) \le c$. Then the deterministic algebraic query complexity of f over F is defined as

$$D_{\mathbb{F},c}(f) = \min_{M \in M} \max_{A,\tilde{A}: mdeg(\tilde{A}) \le c} T_M(\tilde{A})$$

where $T_M(\tilde{A})$ is the number of queries to $\tilde{A}$ made by $M^{\tilde{A}}$. The randomized and quantum algebraic query complexities $R_{\mathbb{F},c}(f)$ and $Q_{\mathbb{F},c}(f)$ are defined similarly, except with (bounded-error) randomized and quantum algorithms in place of deterministic ones.

The Algebraic Query Complexity gives lower bound of $M^{\tilde{A}}$'s queries to solve a f(A) problem.

4.2.1 Multilinear Polynomials

for boolean point z, we define:

$$\delta_z(x) := \prod_{i:z_i=1} x_i \prod_{i:z_i=0} (1 - x_i)$$

to be the unique multilinear polynomial that is 1 at z and 0 elsewhere on the Boolean cube.

For multilinear polynomials we can write them uniquely in the basis of δ_z

$$m(x) = \sum_{z\in\{0,1\}^n} m_z \delta_z(x)$$

$m(z) = m_z$

4.2.2 Lower Bounds by Direct Construction

The paper obtain lower bound by show that there are still flexible boolean point(haven't determine the value or in other word exists several low-degree polynomials that agree on queries but have different value on flexible boolean point) after some queries.

Deterministic Lower Bounds

Lemma 4.13. *Let $\mathbb{F}$ be a field and let $y_1, ..., y_t$ be points in $\mathbb{F}^n$. Then there exists a multilinear polynomial $m : \mathbb{F}^n \to \mathbb{F}$ such that:*

(i) $m(y_i) = 0$ *for* $i \in [t]$

(ii) m(z)=1 for at least $2^n - t$ *boolean points.*

we solve equation for (i), we get at least $2^n - t$ m_z is 1. So at least $2^n - t$ boolean points set to 1.

Lemma 4.14. *Let $\mathbb{F}$ be a field and let $y_1, ..., y_t$ be points in $\mathbb{F}^n$. . Then for at least$2^n - t$ Boolean points w., there exists a multiquadratic extension polynomial p, such that:*

(i) $m(y_i) = 0$ *for* $i \in [t]$

(ii) m(w)=1

(iii) m(z)=0 for all boolean points $z \neq w$.

Using lemma 4.13, we get a multilinear polynomial m has at least $2^n - t$ boolean point set to 1. For a chosen w:

$$p(x) = m(x)\delta_w(x)$$

is the polynomial desired. We denote the polynomial as $p_{w,Y}(x)$ Y is set of point that have been queried. Lemma 4.14 is the key to proof Algebraic Query Complexity. It says that after queries $y_i, i \in [t]$, there are still $2^n - t$ polynomial that are extension of some boolean A_j function that not same on boolean points. Thus the adversary will not figure out the real A.

define OR problem to be whether exists x, s.t. $A(x) = 1$

Theorem 4.15. $D_{\mathbb{F},c}(f) = 2^n$

If $D_{\mathbb{F},c}(f) < 2^n$, we consider the M that solved the problem. After it queries $\tilde{A}$ all points in N $|N| < 2^n$ points, $\tilde{A}$ can either be a extension of 0 or $p_{w,N}$in lemma 4.14, so M can't solve the problem.

Then the paper extend the lemma 4.14 to

Lemma 4.16. *Let F be a collection of fields (possibly with multiplicity). Let f be a Boolean function, and for every $\mathbb{F} \in F$, let $p_{\mathbb{F}} : \mathbb{F}^n \to \mathbb{F}$ be a multiquadratic polynomial over $\mathbb{F}$ extending f. Also let $Y_{\mathbb{F}} \subset \mathbb{F}^n$ for each $\mathbb{F} \in C$, and $t := \sum_{\mathbb{F} \in C} |Y_{\mathbb{F}}|$ Then there exists a subset $B \subset \{0,1\}^n$, with $|B| \leq t$, such that for all Boolean functions f' f on B, there exist multiquadratic polynomials $p'_{\mathbb{F}} : \mathbb{F}^n \to \mathbb{F}$ (one for each $\mathbb{F} \in F$) such that:*

(i) $p'_{\mathbb{F}}$ extends f', and

(ii) $p'_{\mathbb{F}}(y) = p_{\mathbb{F}}(y)$ for all $y \in Y_{\mathbb{F}}$

This extends lemma 4.14 from one field to collection of fields, that even if M is query in different f's polynomial extensions on different fields there is still many choices of f's actual value.This lemma also shows the multiquadratic polynomials can be any boolean value on at least $2^n - t$ boolean points.

The result is obtained by considering good points: the point w that can generate a $p_{w,Y}$ in lemma 4.14.

$$p'_{\mathbb{F}}(x) = p_{\mathbb{F}}(x) + \sum_{w \text{ is good point}} (f'(x) - f(x)) p_{w,Y_{\mathbb{F}}}(x)$$

In this case $p_{w,Y_{\mathbb{F}}}$ is like a basis.

Probabilistic Lower Bounds The lower bound of randomized algorithm will be done via the Yao minimax principle[8]: the worse cost of randomized algorithm consider all possible input is not smaller than the average cost of the best deterministic machine on the random input determined by distribution.

$$\max_{x \in \mathcal{X}} E[c(A, x)] \geq \min_{a \in \mathcal{A}} E[c(a, X)]$$

$\mathcal{X}$ is input distribution, X is a random input,x is a input. $\mathcal{A}$ is a collection of deterministic algorithms, A is a random algorithms.

The proof in the paper consider a uniform distribution on polynomials, so it can't be use in integer case(finite field extends to integer)

Lemma 4.17. *Let $\mathbb{F}$ be a finite field. Also, for all w , let D_w be the uniform distribution over multiquadratic polynomials p such that p (w) = 1 and p (z) = 0 for all Boolean $z \neq w$. Suppose an adversary chooses a "marked point" w uniformly at random, and then chooses p according to D_w. Then any deterministic algorithm, after making t queries to p,will have queried w with probability at most $t/2^n$.*

the proof of lemma 4.17 consist of two parts. First show after t queries, the set of possible p is not two small, this can be done using lemma 4.14 we can show that it related to number of good points. Next show that after t queries the possibility of p is in D_x is same as D'_x, this can be done by giving a bijection

$$v \in V_x \iff v - u_x + u'_x \in V_{x'}$$

V_w denote $p \in D_w$ that can satisfy the first t queries.

Combine the two result we know that every times M queries, it is a random query and in each query the it cannot eliminate too many possible of which set D_w he is querying (which w is chosed)

An immediate corollary is

Theorem 4.18. *$R_{\mathbb{F},c}(OR) = \Omega(2^n)$for every finite field $\mathbb{F}$*

Lemma 4.19. *Let $\mathbb{F}$ be a finite field. Also, for all w , let $D_{w,\mathbb{F}}$ be the uniform distribution over multiquadratic polynomials p such that p (w) = 1 and p (z) = 0 for all Boolean $z \neq w$. And p is over $\mathbb{F}$Suppose an adversary chooses a "marked point" w uniformly at random. Then for every finite field $\mathbb{F}$ choose $p_{\mathbb{F}}$ according to $D_{w,\mathbb{F}}$. Then any deterministic algorithm, after making t queries to any combination of $p_{\mathbb{F}}$,will have queried w with probability at most $t/2^n$.*

This is a extension of lemma 4.17, allowing adversary to query any combination of $p_{\mathbb{F}}$

4.2.3 Lower Bounds by Communication Complexity

Theorem 4.20 (transfer principle)**.** *Let A be a Boolean function, and let $\tilde{A}$: be the unique multilinear extension of A over a finite field F. Suppose one can evaluate some Boolean predicate f of A using T deterministic adaptive queries to $\tilde{A}$. Also, let A_0 and A_1 be the subfunctions of A obtained by restricting the first bit to 0 or 1 respectively. Then if Alice is given the truth table of A_0 and Bob is given the truth table of A_1, they can jointly evaluate f(A) using O(Tnlog|F|) bits of communication.*

This is proved by let Alice and Bob simulate the machine M that evaluates f using T queries to $\tilde{A}$.

$$\tilde{A}(x) = \sum_{z\in\{0,1\}^n} A(z)\delta_z(x)$$

$$\tilde{A}(x) = \sum_{0z\in\{0,1\}^{n-1}} A(0z)\delta_{0z}(x) + \sum_{1z\in\{0,1\}^{n-1}} A(1z)\delta_{1z}(x)$$

So alice can compute the left term L_A and send to Bob $< x, L_A >$ using $O(nlog|\mathbb{F}|)$ bit, then Bob can calculate $\tilde{A}(x)$ and by simulating M, he can determine what to query. So totally $O(Tnlog|\mathbb{F}|)$ bits.

Notice that if M is random or quantum, we will random ro quantum communication protocol, thus the transfer principle can extends to different computational model.

We define DISJ problem to be for $A : \{0,1\}^n \to \{0,1\}$, is there a $x \in \{0,1\}^{n-1}$, s.t. A(0x)=A(1x).

Theorem 4.21. $R_{\mathbb{F},c}(DISJ) = \Omega(\frac{2^n}{nlog|\mathbb{F}|})$ *for every finite field* $\mathbb{F}$

The result is obtained by seeing DISJ problem as the Disjointness problem of the truth table of A_0 and A_1.

4.3 The Need for Non-Algebrizing Techniques

4.3.1 Non-Algebrizing Techniques Needed for P vs. NP

Theorem 4.22. *There exist* $A, \tilde{A}$ *such that* $NP^{\tilde{A}} \subset P^A$

Let A be a PSPACE-complete language, then by Babai et al.[7],the multilinear extension of any PSPACE language is also in PSPACE, so $\tilde{A}$ is a PSPACE-complete language, thus we proof the theorem. And also implies:

Theorem 4.23. *There exist* $A, \tilde{A}$ *such that* $PSPACE^{\tilde{A}[poly]} \subset P^A$

Theorem 4.24. *There exist* $A, \tilde{A}$ *such that* $NP^A \not\subset P^{\tilde{A}}$ *. Furthermore, the language L that achieves the separation simply corresponds to deciding, on inputs of length n, whether there exists a* $w \in \{0,1\}^n$ *with* $A_n(w) = 1$*.*

We proof the language L is not in $DTIME(n^{logn})^{\tilde{A}}$. We use diagonalization: assign each $M_i^{\tilde{A}}$ a n_i. Because the lemma 4.14, after less than 2^n there exists some good point w, so if $M_i^{\tilde{A}}(n_i) = 1$ set $A_{n_i} = 0$ or set $A_{n_i} = p_{w,Y}$.Thus we construct $\tilde{A}$ that achieves seperation.

And if we replace $2^n > n^{logn}$ to $2^{n-1} > n^{logn}$, we can proof the result for RP.

4.3.2 Non-Algebrizing Techniques Needed for NP vs. BPP

Theorem 4.25. *There exist* $A, \tilde{A}$ *such that* $NP^A \not\subset BPP^{\tilde{A}}$ *Furthermore, the language L that achieves the separation simply corresponds to finding a* $w \in \{0,1\}^n$ *with* $A_n(w) = 1$*.*

The proof is similar to Bennett and Gill's[9]

We generate the oracle by, for each $n \in \mathbb{N}$ (1) randomly draw w_n, (2) $A_n(w) = 1$ and =0 in other boolean points. Next, for every finite field $\mathbb{F}$, draw the extension $\tilde{A}_{n,\mathbb{F}}$ of A_n from $D_{n,w_n,\mathbb{F}}$.

We define the language L as follows: $0^i1^{n-i} \in L$ iff. the ith bit of w_n is a 1, and $x \notin L$ for all x not of the form 0^i1^{n-i}.

Intuitivly if BPP machine M can decides $0^i1^{n-i} \in L$ for all i with high probability. It decides w_n, contradict to lemma 4.17.

Theorem 4.26. *There exist $A, \tilde{A}$ such that $NP^A \not\subset P^{\tilde{A}}/poly$.*

If this is not true, then $BPP^{\tilde{A}}$ can guess the advice and contradict to the result in theorem 4.25.

4.3.3 Non-Algebrizing Techniques Needed for Circuit Lower Bounds

Theorem 4.27. *There exist $A, \tilde{A}$ such that $NTIME^{\tilde{A}}(2^n) \not\subset SIZE^A(n)$.*

We introduce the key idea of this proof in steps:

- First we focus on $M_1, \cdots, M_n$(enumerations of oracle turning machine in $NTIME^{\tilde{A}}(2^n)$) when input size is n, and only allow M to query a single polynomial $p : \mathbb{F}^{4n} \to \mathbb{F}$
- Second we force <i,x> to accept.

  We define a few concept for the sake of discussion.

  - A point y is active if p(y) haven't be determined. y is inactive if p(y) is assign a value and cannot be changed.
  - <i,x> is the result of $M_i(x)$. $|x| = n, i \leq n$
  - <i,x> accept is to say <i,x>=1.
  - fix a point y is making it from active to inactive by assign a value $c_y = p(y)$

  We try to make all <i,x> accept by fix point they queries. We do this by enumeration of all <i,x>'s. If <i,x> does accept, fixs all point on computational path output 1.

- We at most make $n2^n$ <i,x> accept, leading to at most $n2^n2^n$ inactive points, lefting at least $2^{4n} - n2^{2n}$ good points. Using the lemma 4.16, we assign them any boolean value.
- assoiate <i,x> with a unique 4n bits long string $w_{i,x}$ .using union bound we can show exists a z', s.t.

  $$z' \oplus w_{i,x} \text{ is a good point}$$

  $\forall w_{i,x}$. Now we can assign $f(z' \oplus w_{i,x}) =< i, x >$, the new assignment of f will not lead to new acceptance of <i,x>, so it is correct. And by computing $z' \oplus w_{i,x}$ linear-size circuit can simulate $< i, x >$.

- extends it to leting oracle machine query combination of polynomial over different fields
- Including: First we force all <i,x> to stable, in case of Nondeterministic machine, accept, and they encode the result of <i,x> on the good point, give circuit with oracle A the ability to output oracle machine's result.

By a standard padding argument, Theorem 4.27 immediately gives $A, \tilde{A}$ such that $NEXP^{\tilde{A}} \not\subset P^A/poly$

Theorem 4.28. *There exist $A, \tilde{A}$ such that $EXP^{NP^{\tilde{A}}} \not\subset P^A/poly$*

Theorem 4.29. *There exist $A, \tilde{A}$ such that $BPEXP^{\tilde{A}} \not\subset P^A/poly$*

The paper omit the proof of these two theorems, since they have similar idea to 4.27

4.3.4 Non-Algebrizing Techniques Needed for Other Problems

We can use the communication complexity transfer principle from to achieve many other separations

Theorem 4.30. *There exist $A, \tilde{A}$ such that*

- $NP^A \not\subset BPP^{\tilde{A}}$
- $coNP^A \not\subset MA^{\tilde{A}}$
- $P^{NP^A} \not\subset PP^{\tilde{A}}$
- $NP^A \not\subset BQP^{\tilde{A}}$
- $BQP^A \not\subset BPP^{\tilde{A}}$
- $QMA^A \not\subset MA^{\tilde{A}}$

We omit the proof of it here, and explain the general idea of using transfer principle.

A is a complexity class, then A_{cc} is define to be the set of language that can be solve by two A machine using $O(polylogN)$ communication. If $C^A \subset D^{\tilde{A}}$ for all $A, \tilde{A}$, then $C_{cc} \subset D_{cc}$, for D that can only queries poly(n) times, and C that any problem f(A) is in C^A. If not exist $L \in C_{cc}$ $L \notin D_{cc}$. We construct the oracle A by seeing the two input of D_{cc} is truth table of A_0, A_1, and generate a question L' with input 1^{logN} and ask same question as L. If $L' \in D^{\tilde{A}}$ then $T \leq poly(logN)$ so $L' \notin D^{\tilde{A}}$, but $L' \in C^A$, thus gives contradiction. So we can get separation result using separation result of $C_{cc} \not\subset D_{cc}$.

5 conclusion

In this paper Aaronson, Scott and Avi Wigderson. gives a more broad barrier for complexity theory: Algebrization. And proved result that algebrize and result that does not. Also they give method to proof both cases.

We can see actually this gives us a intuition on how to study a proof technique T and its barrier.

- First we generate a group of result A' correspond to A$C \subset D$ or other realtionship between complexity classes. A' should be proof using the same structure and same technique in proving A. And using A' and A we may generate a kind of relativizing property, denote as T-izing, that is guaranteed if one uses T technique.
- For proving A result T-izes, we proof it using the proof of A. For proving A result non-T-izes. We will facing constructing a special environment like oracle such that A does not hold. They will still face the problem like oracle separation and need to compute query complexity. At last using the part in oracle haven't been determine to obatin result we want.

Furthermore,because some class with oracle is equal to some class without oracle.($P^{PSPACE} = PSPACE$). Can we view some result A that has no oracle as a result other result B relatives,algebrizes,and etc, thus gives such barrier a more broader usage.

6 Acknowledgment

I would like to express my gratitude to Professor Penghui Yao and Teaching Assistant Xudong Wu for delivering such an engaging class on Computational Complexity. This course has significantly broadened my understanding of computation and provided me with insights into how the world operates. It has been a truly enriching experience, expanding my perspective on the intricacies of computational theory.

References

[1] Aaronson, Scott and Avi Wigderson. “Algebrization: A New Barrier in Complexity Theory.” ACM Trans. Comput. Theory 1 (2009): 2:1-2:54.

[2] T. Baker, J. Gill, and R. Solovay. Relativizations of the P=?NP question. SIAM J. Comput., 4:431–442, 1975.

[3] A. A. Razborov and S. Rudich. Natural proofs. J. Comput. Sys. Sci., 55(1):24–35, 1997.

[4] C. Lund, L. Fortnow, H. Karloff, and N. Nisan. Algebraic methods for interactive proof systems. J. ACM, 39:859–868, 1992.

[5] A. Shamir. IP=PSPACE. J. ACM, 39(4):869–877, 1992.

[6] A. Juma, V. Kabanets, C. Rackoff, and A. Shpilka. The black-box query complexity of polynomial summation. ECCC TR07-125, 2007.

[7] L. Babai, L. Fortnow, and C. Lund. Nondeterministic exponential time has two-prover interactive protocols. Computational Complexity, 1(1):3–40, 1991

[8] https://en.wikipedia.org/wiki/Yao

[9] C. H. Bennett and J. Gill. Relative to a random oracle A, P A 6= NP A 6= coNP A with probability 1. SIAM J. Comput., 10(1):96–113, 1981.