Derandomizing Polynomial Identity Tests Means Proving Circuit Lower Bounds

Devansh Shringi Rishabh Batra
CS640 Project

October 6, 2021

Contents

Introduction

- We present a summarized interpretation of the paper Derandomizing Polynomial Identity Tests Means Proving Circuit Lower Bounds [KI03] by Valentine Kabanets and Russell Impagliazzo, based on our reading of the paper.
- Following is the main theorem of the paper

Theorem

$$PIT \in P \implies per \notin Arth - P/poly \ or \ NEXP \not\subseteq P/poly$$

Structure

- **Preliminaries:** Arithmetic circuits, PIT, PRGs
- **Lemma 1**
  $PIT \in P$ and $per \in Arth - P/poly \implies P^{per} \subseteq NP$.
- **Lemma 2** $EXP \subseteq P/poly \implies EXP = MA$
- **Lemma 3** $NEXP \in P/poly \implies NEXP = EXP$.
- **Proof of Theorem:** Combining to get the main theorem.
- **Conclusion:** Implications and Future Scope

Arithmetic circuits

- Representation for polynomials
- A Directed Acyclic Graph that computes a polynomial f over $\mathbb{F}$ and set of variables $x_1, \ldots, x_n$
- Vertices of in-degree 0 labeled with variable or field element
- All other vertices(gates) labeled with $+$ or $\times$
- Edges labeled with field constants (1 by default)
- **Size:** number of edges
- For more details on Arithmetic circuits, check [SY10]

Arithmetic circuits

Example

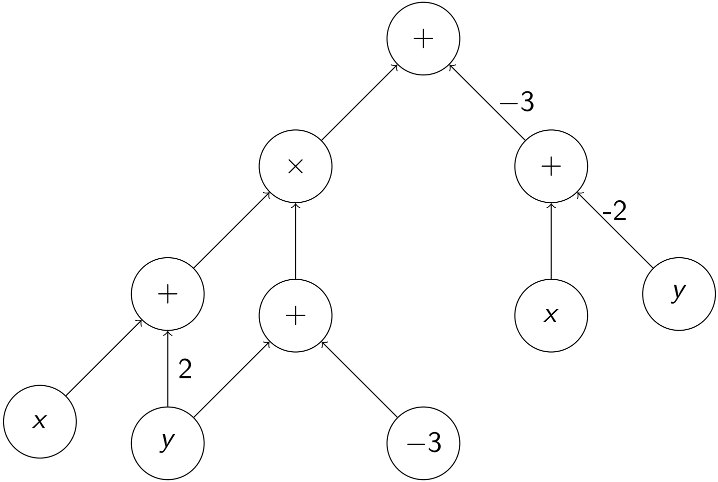

Figure: Circuit computing $xy + 2y^2$

Polynomial Identity Testing(PIT)

- Efficiently test whether an input polynomial as circuit is identically zero or not.
- For univariate, just check at $\textit{degree} + 1$ points. Doesn't work for multivariate.
- For more details on PIT, check [Sax09]

Randomized Solution

Lemma

(PIT Lemma)*(Schwartz-Zippel[Sch80]) Let $f \in \mathbb{F}[x_1, \ldots, x_n]$ be a non-zero polynomial of total degree $d \geq 0$. Let S be any finite subset of $\mathbb{F}$, and let $\alpha_1, \ldots, \alpha_n$ be elements selected independently, uniformly and randomly from S. Then,*

$$Pr_{\alpha_1,\ldots,\alpha_n \in S}[f(\alpha_1, \ldots, \alpha_n) = 0] \leq \frac{d}{|S|}$$

Randomized Solution

Lemma

(PIT Lemma)*(Schwartz-Zippel[Sch80]) Let $f \in \mathbb{F}[x_1, \ldots, x_n]$ be a non-zero polynomial of total degree $d \geq 0$. Let S be any finite subset of $\mathbb{F}$, and let $\alpha_1, \ldots, \alpha_n$ be elements selected independently, uniformly and randomly from S. Then,*

$$Pr_{\alpha_1,\ldots,\alpha_n \in S}[f(\alpha_1, \ldots, \alpha_n) = 0] \leq \frac{d}{|S|}$$

- Thus $PIT \in coRP$
- *Open:* Derandomizing PIT in $poly(s)$-time

Pseudorandomness Generators(PRGs)

- Decrease the number of random bits required.
- For $S : \mathbb{N} \to \mathbb{N}$, a $2^{O(n)}$-computable function $G : \{0,1\}^* \to \{0,1\}^*$ is an $S-prg$, if $\forall l$, $G : \{0,1\}^l \to \{0,1\}^{S(l)}$, and $\forall$ circuits C of size $\leq S(l)^3$

$$|Pr_{x \in U_l}[C(G(x)) = 1] - Pr_{x \in U_{S(l)}}[C(x) = 1]| < 0.1$$

Pseudorandomness Generators(PRGs)

- Decrease the number of random bits required.
- For $S : \mathbb{N} \rightarrow \mathbb{N}$, a $2^{O(n)}$-computable function $G : \{0,1\}^* \rightarrow \{0,1\}^*$ is an $S - prg$, if $\forall l$, $G : \{0,1\}^l \rightarrow \{0,1\}^{S(l)}$, and $\forall$ circuits C of size $\leq S(l)^3$

$$|Pr_{x \in U_l}[C(G(x)) = 1] - Pr_{x \in U_{S(l)}}[C(x) = 1]| < 0.1$$

- If a S-prg exists then $\forall$ functions l

$$BP - TIME(S(l(n))) \subseteq DTIME(2^{l(n)}S(l(n)))$$

- A $2^{\epsilon l}$-prg $\implies$ BPP=P

Hardness

- **Worst-case Hardness** For $f : \{0,1\}^* \rightarrow \{0,1\}$, $H_{wrs}(f)$ is the largest $S(n)$ st. $\forall$ circuit $C_n \in size(S(n))$,

$$Pr_{x \in U_n}[C_n(x) = f(x)] < 1$$

- **Average-case Hardness** $H_{avg}(f)$ is the largest $S(n)$ st. $\forall$ circuit $C_n \in size(S(n))$,

$$Pr_{x \in U_n}[C_n(x) = f(x)] < \frac{1}{2} + \frac{1}{S(n)}$$

- Can be shown that a worst-case hard function gives also an average-case hard function.

NW-Design

Theorem

If $\exists f \in E$ with $H_{avg} \geq S(n)$, then $\exists S'(l)$-prg, where $S'(l) = S(n)^{0.01}$.

Lemma 1

Lemma

$PIT \in P$ *and* $per \in Arth - P/poly \implies P^{per} \subseteq NP.$

Proof Idea

- "Guess" the small circuit for permanent and verify it using $PIT \in P$.
- $per_n(A) = \sum_{i \in [n]} A_{1i}.per(A'_{1i})$ where A'_{1i} is the corresponding minor.

- Let C_n be arithmetic circuit corresponding to the per_n.
- Protocol for obtaining the circuit.
  1. Given C_{n-1}, we guess the circuit for C_n as follows:

$$C_n(A) = \sum_{i \in [n]} A_{1i}.C_{n-1}(A'_{1i}) \ldots \ldots (1)$$

  2. Use PIT for verifying whether the above expression is correct or not.
  3. Repeat it for circuits C_{n-1} which we used for minors and so on.
- Using this recursive guess and verify procedure, we can get a circuit $C_n(A) = per_n(A)$ by induction on n.

- Now we show $P^{per} \subseteq NP$
- Let $L \in P^{per}$.
  Guess C_n for per_n using the recursive procedure.
  Use this circuit C_n for per_n instead of the oracle
- $PIT \in P$, implies the entire verification is in P.
- $per \in Arth - P/poly$, implies the guess that our machine need to do is poly-sized.
- This gives $L \in NP \implies P^{per} \subseteq NP$

Lemma 2

Lemma

$EXP \subseteq P/poly \implies EXP = MA$

Proof Idea First show $EXP \subseteq P/poly \implies EXP = \Sigma_2$.

- Consider $L \in EXP$, with TM N. Encode steps of N Using the circuit and $\exists\forall$
- Compute j-th bit of i-th configuration of $N(x)$ in exp-time $\implies \exists$ poly-size $C(x, i, j)$ computing it.
- $x \in L \iff \exists C, \forall(i, j)[C(x, i, j) \to C(x, i+1, j)$ is a valid step].
- Thus, $EXP = \Sigma_2$

Lemma 2

Lemma

$EXP \subseteq P/poly \implies EXP = MA$

Proof Idea contd.

- $\Sigma_2 \subseteq PSPACE = IP \subseteq EXP = \Sigma_2$, i.e. $PSPACE = IP = EXP \subseteq P/poly$.
- We have a IP protocol for L. We convert it one round.
- Prover in IP is a PSPACE machine, simulate using a poly-size circuit family $\{C_n\}_{n \in \mathbb{N}}$
- 1-round protocol for checking $x \in L$:
  Prover: Send his circuit C_n, for $n = |x|$.
  Verifier: Simulate the IP protocol using C_n as P.
- Thus, $EXP = MA$

Lemma 3

Lemma

$NEXP \subseteq P/poly \implies NEXP = EXP$

Proof Idea

- Assume $\exists L \in NEXP \setminus EXP$, st. $\exists c > 0$ and machine $R(x, y)$ running in $exp(|x|^{10c})$

$$x \in L \iff \exists y \in \{0,1\}^{exp(|x|^c)} R(x,y) = 1$$

- y is hard for EXP,we use it compute hard-function, consider function whose Truth table is y.

Lemma 3

Lemma

$NEXP \subseteq P/poly \implies NEXP = EXP$

Proof Idea contd.

Consider the machine M_D, $\forall D > 0$ as follows:

- construct tt of all circuits of size n^{100D}, with n^c input.
- if $\exists C, R(x, tt) = 1$ ACCEPT, else REJECT

Running Time: $exp(n^{100D} + n^{10c})$

Lemma 3

Lemma

$NEXP \subseteq P/poly \implies NEXP = EXP$

Proof Idea contd.

- $L \notin EXP \implies M_D$ cannot solve L
- Therefore, for infinitely many x's, y is such that $H_{wrs}(f_y) > n^{100D}$.
- Using NW design we have a l^D prg.

Lemma 3

Lemma

$NEXP \subseteq P/poly \implies NEXP = EXP$

Proof Idea contd.

- $EXP \subset NEXP \subseteq P/poly$. So from lemma 2, we have an EXP=MA
- $\forall L \in EXP$, Merlin tries to show that $x \in L$ by sending a short proof to Arthur.
- Arthur verifies it using a randomised algo in say n^D steps.
- Using the I^D prg, we can reduce the number of random bits from n^D to n for Arthur.

Lemma 3

Lemma

$NEXP \subseteq P/poly \implies NEXP = EXP$

Proof Idea contd.

- If we have n as the input length of some string which is "hard" for the tt circuits, we can replace Arthur by a non-deterministic algorithm in poly$(n^d)2^{n^c}$ time that does not toss any random coins by using the prg obtained before (the 2^{n^c} factor is for calculating the n random bits deterministically)
- This gives $L \in$ NTIME $(2^{n^{c'}})$ "infinitely often" with n-bit advice. Thus $EXP \subseteq$ NTIME $(2^{n^{c'}})$ "infinitely often" with n-bit advice
- But $NEXP \subseteq P/poly$. Thus we have NTIME $(2^{n^{c'}})$ $\subseteq SIZE(n^{c'})$ for a constant c'. So $EXP \subseteq SIZE(n^{c'})$ infinitely often (the n bit advice can be hardcoded in the circuit).

Lemma 3

Lemma

$NEXP \subseteq P/poly \implies NEXP = EXP$

Proof Idea contd.

- $\exists$ c' such that every language in EXP can be decided on infinitely many inputs by a circuit family of size $n + n^{c'}$. Yet this can be ruled out using elementary diagonalization (more details in the paper)

Proof of Theorem

Theorem

$$PIT \in P \implies per \notin Arth - P/poly \text{ or } NEXP \not\subseteq P/poly$$

- Suppose $PIT \in P$, $per \in Arth - P/poly$ and $NEXP \subseteq P/poly$.
- From lemmas 2 and 3,$NEXP = EXP = MA \subseteq PH$.
- Also $PH \subseteq P^{per}$ (Toda's theorem)
- So $NEXP \subseteq P^{per}$
- Now as we have $PIT \in P$ and $per \in Arth - P/poly$, using lemma 1, we get $P^{per} \subseteq NP$
- Combining these two, we get $NEXP \subseteq NP$ which contradicts the non-deterministic time hierarchy theorem. Thus atleast of the assumptions is false which gives:

$$PIT \in P \implies per \notin Arth - P/poly \text{ or } NEXP \not\subseteq P/poly$$

Conclusion

- Derandomizing RP or BPP would give us circuit lower bounds for NEXP or for permanent.
- The results in the present paper do not rule out that ZPP $=$ P can be proved without having to prove any circuit lower bounds first. This leaves some hope that unconditional derandomization of ZPP could be achieved.

Open Problems

- $BPP = P$, $PIT \in P$, $per \notin Arth - P/poly$ and $NEXP \not\subseteq P/poly$.(we believe all of these to be true)
- Does BPP=P imply circuit lower bounds for EXP (instead of NEXP) ?

Questions

Questions ?

References I

Valentine Kabanets and Russell Impagliazzo.
Derandomizing polynomial identity tests meansproving circuit lower bounds.
ACM symposium on Theory of computing, 2003.

Nitin Saxena.
Progress on polynomial identity testing.
Bulletin of the EATCS, 99:49–79, 2009.

Jacob T Schwart.
Fast probabilistic algorithms for verification of polynomial identities.
Journal of the ACM (JACM), 1980.

References II

Amir Shpilka and Amir Yehudayoff.
Arithmetic circuits: A survey of recent results and open questions.
Foundations and Trends in Theoretical Computer Science: Vol. 5, 2010.