New Algebrization Barriers to Circuit Lower Bounds via Communication Complexity of Missing-String

Lijie Chen
University of California, Berkeley, CA, USA

Yang Hu
Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, China

Hanlin Ren
Institute for Advanced Study, Princeton, NJ, USA

Abstract

The *algebrization barrier*, proposed by Aaronson and Wigderson (STOC '08, ToCT '09), captures the limitations of many complexity-theoretic techniques based on arithmetization. Notably, several circuit lower bounds that overcome the relativization barrier (Buhrman–Fortnow–Thierauf, CCC '98; Vinodchandran, TCS '05; Santhanam, STOC '07, SICOMP '09) remain subject to the algebrization barrier.

In this work, we establish several new algebrization barriers to circuit lower bounds by studying the communication complexity of the following problem, called XOR-Missing-String: For $m < 2^{n/2}$, Alice gets a list of m strings $x_1, \ldots, x_m \in \{0,1\}^n$, Bob gets a list of m strings $y_1, \ldots, y_m \in \{0,1\}^n$, and the goal is to output a string $s \in \{0,1\}^n$ that is not equal to $x_i \oplus y_j$ for any $i, j \in [m]$.

1. We construct an oracle A_1 and its multilinear extension $\widetilde{A_1}$ such that $\mathsf{PostBPE}^{\widetilde{A_1}}$ has linear-size A_1-oracle circuits on infinitely many input lengths. That is, proving $\mathsf{PostBPE} \not\subseteq \text{i.o.-}\mathsf{SIZE}[O(n)]$ requires non-algebrizing techniques. This barrier follows from a PostBPP communication lower bound for XOR-Missing-String. This is in contrast to the well-known algebrizing lower bound $\mathsf{MA_E}\ (\subseteq \mathsf{PostBPE}) \not\subseteq \mathsf{P}/_{\mathrm{poly}}$.
2. We construct an oracle A_2 and its multilinear extension $\widetilde{A_2}$ such that $\mathsf{BPE}^{\widetilde{A_2}}$ has linear-size A_2-oracle circuits on all input lengths. Previously, a similar barrier was demonstrated by Aaronson and Wigderson, but in their result, $\widetilde{A_2}$ is only a *multiquadratic* extension of A_2. Our results show that communication complexity is more useful than previously thought for proving algebrization barriers, as Aaronson and Wigderson wrote that communication-based barriers were "more contrived". This serves as an example of how XOR-Missing-String forms new connections between communication lower bounds and algebrization barriers.
3. Finally, we study algebrization barriers to circuit lower bounds for $\mathsf{MA_E}$. Buhrman, Fortnow, and Thierauf proved a *sub-half-exponential* circuit lower bound for $\mathsf{MA_E}$ via algebrizing techniques. Toward understanding whether the half-exponential bound can be improved, we define a natural subclass of $\mathsf{MA_E}$ that includes their hard $\mathsf{MA_E}$ language, and prove the following result: For every *super-half-exponential* function $h(n)$, we construct an oracle A_3 and its multilinear extension $\widetilde{A_3}$ such that this natural subclass of $\mathsf{MA_E}^{\widetilde{A_3}}$ has $h(n)$-size A_3-oracle circuits on all input lengths. This suggests that half-exponential might be the correct barrier for $\mathsf{MA_E}$ circuit lower bounds w.r.t. algebrizing techniques.

2012 ACM Subject Classification Theory of computation → Complexity classes; Theory of computation → Circuit complexity; Theory of computation → Communication complexity

Keywords and phrases circuit lower bound, algebrization barrier, missing string, communication complexity

Digital Object Identifier 10.4230/LIPIcs.ITCS.2026.37

Related Version *Full Version*: https://eccc.weizmann.ac.il/report/2025/184/

1 Introduction

Proving unconditional circuit lower bounds is one of the major challenges in theoretical computer science, with the holy grail of proving $\mathsf{NP} \not\subseteq \mathsf{P}/_{\mathrm{poly}}$. Unfortunately, our progress toward this goal is barely satisfactory, as it is even open to prove a super-polynomial size lower bound for huge, exponential-time classes such as NEXP. Only for even larger complexity classes such as $\Sigma_2\mathsf{EXP}$ [27], which are in (or beyond) the second level of exponential-time hierarchy, are super-polynomial lower bounds known.

Our lack of progress in proving circuit lower bounds is partially explained by a series of *barrier results* such as relativization [10], natural proofs [37], and algebrization [3]. Among these barriers, relativization and algebrization are particularly relevant to lower bounds against *unrestricted circuits* for *large* complexity classes. For example, Wilson [44] constructed an oracle world where P^{NP} has linear-size circuits, which explains our inability to prove fixed-polynomial size lower bounds for P^{NP}.

Missing-String: a duality-based approach to relativizing circuit lower bounds. Previous relativization barriers for circuit lower bounds are proved in an *ad hoc* fashion, which involves carefully analyzing the interaction between the oracle and the machines to be diagonalized [44, 12, 2]. A recent paper by Vyas and Williams [43] introduced the Missing-String problem as a systematic approach to such relativization barriers:

▶ **Problem 1** (Missing-String(n, m))**.** *Let n, m be such that $m < 2^n$. Given (query access to) a list of length-n strings $x_1, x_2, \ldots, x_m$, output a length-n string that is not in this list.*

The query complexity of Missing-String captures relativizing circuit lower bounds in the following sense: relativization barriers to proving $\mathcal{C}\text{-}\mathsf{EXP} \not\subseteq \mathsf{P}/_{\mathrm{poly}}$ are essentially $\mathcal{C}^{\mathrm{dt}}$ lower bounds for Missing-String. (Here, $\mathcal{C}\text{-}\mathsf{EXP}$ is the exponential-time version of $\mathcal{C}$ and $\mathcal{C}^{\mathrm{dt}}$ means the decision-tree version of $\mathcal{C}$.) This connection was made explicit in [43], who demonstrated an equivalence between relativization barriers to exponential lower bounds for $\Sigma_2\mathsf{E}$ and the non-existence of small depth-3 circuits for Missing-String.[1]

Intriguingly, Missing-String connects *circuit lower bounds* with *circuit upper bounds* and provides a *duality*-based approach to both: Decision tree upper bounds for Missing-String imply relativizing circuit lower bounds, and decision tree lower bounds for Missing-String imply relativized worlds with circuit upper bounds (i.e., relativization barrier to circuit lower bounds). Besides providing a clean and systematic method for relativization barriers, this "algorithmic" perspective has indeed made progress in circuit lower bounds: By designing algorithms for the *Range Avoidance* problem [30, 32, 38], which is the "white-box" version of Missing-String,[2] recent work [15, 33] proved an exponential-size, relativizing lower bound for the complexity class $\Sigma_2\mathsf{E}$ (which also implies a quasi-polynomial size depth-3 circuit upper bound for Missing-String, settling the question in [43]).

The quest of algebrization. However, the relativization barrier does not capture many circuit lower bounds that follow from *nonrelativizing* results such as $\mathsf{IP} = \mathsf{PSPACE}$ [35, 40] and $\mathsf{MIP} = \mathsf{NEXP}$ [8, 9], whose proofs are based on nonrelativizing proof techniques such as *arithmetization*. This includes circuit lower bounds for PP [42] and MA [12, 39], both of which are *provably nonrelativizing* [12, 2].

---

[1] $\mathsf{E} = \mathsf{DTIME}[2^{O(n)}]$ denotes *single-exponential* time and $\mathsf{EXP} = \mathsf{DTIME}[2^{n^{O(1)}}]$ denotes *exponential time*; classes such as $\Sigma_2\mathsf{E}$ and $\Sigma_2\mathsf{EXP}$ are defined analogously. Exponential time and single-exponential time are basically interchangeable in the context of super-polynomial lower bounds by a padding argument.

[2] In the Range Avoidance problem, we are given the description of a circuit $C : \{0,1\}^n \to \{0,1\}^{n+1}$ and our goal is to output a string $y \in \{0,1\}^{n+1}$ that is not in the range of C. It is easy to see that *relativizing* algorithms for Range Avoidance are equivalent to decision trees for Missing-String.

To shed light on the limitations of such proof techniques, Aaronson and Wigderson [3] proposed the *algebrization* barrier: a lower bound statement $\mathcal{C} \not\subseteq \mathcal{D}$ *algebrizes* if $\mathcal{C}^{\widetilde{A}} \not\subseteq \mathcal{D}^{A}$ for all oracles A and all low-degree extensions $\widetilde{A}$ of A. Aaronson and Wigderson showed that many arithmetization-based results indeed algebrize; in particular, $\mathsf{MA}_{\mathsf{E}}^{\widetilde{A}} \not\subseteq \mathsf{P}^{A}/_{\mathrm{poly}}$ for every oracle A and low-degree extension $\widetilde{A}$ of A [3, Theorem 3.17]. In fact, all super-polynomial lower bounds against general circuits that we are aware of are algebrizing. Thus, algebrization is a more suitable framework than relativization for capturing the limitations of our current techniques.

Unfortunately, our understanding of the algebrization barrier remains primitive. For instance, several aspects of the algebrizing lower bound $\mathsf{MA}_{\mathsf{E}} \not\subseteq \mathsf{P}/_{\mathrm{poly}}$ [12] remain poorly understood even with respect to algebrizing techniques:

- This lower bound only holds *infinitely-often*. That is, [12] only exhibited a language $L \in \mathsf{MA}_{\mathsf{E}}$ where there are *infinitely many* input lengths on which L has high circuit complexity. Recently, the relativizing infinitely-often lower bound for $\Sigma_2\mathsf{E}$ was improved to almost everywhere by [33], and it is tempting to ask whether the same improvement can be made with respect to this lower bound. Does MA_{E} require large circuit complexity on *every* input length? If so, can we show this via algebrizing proof techniques?
- This lower bound is only *sub-half-exponential*. Roughly speaking, a function h is *half-exponential* if $h(h(n)) \approx 2^n$. Half-exponential bounds appear naturally in many win-win analyses in complexity theory [36], and [12] is no exception – it is only known how to prove a size-$h(n)$ lower bound for MA_{E} when h is smaller than half-exponential. The recent "iterative win-win paradigm" [17, 15, 16] provides new techniques for overcoming the half-exponential barrier, which has indeed improved the circuit lower bound for $\Sigma_2\mathsf{EXP}$ to a near-maximum $(2^n/n)$ one [15]. Can we use similar techniques to prove an exponential size lower bound for MA_{E}? If so, can we show this via algebrizing proof techniques?

Our lack of understanding on algebrization barriers raises the following question:

Is there a *duality-based approach* to algebrization barriers?

In particular, can we establish algebrization barriers via lower bounds for Missing-String? It follows from [3] that *communication* lower bounds for Missing-String imply algebrization barriers to circuit lower bounds.[3] Hence, a naïve attempt is to prove communication lower bounds for Missing-String and translate them into algebrization barriers. Unfortunately, it turns out that Missing-String admits an efficient deterministic communication protocol by a simple binary search.[4]

The main conceptual contribution of this paper is the following communication problem:

▶ **Problem 2** (XOR-Missing-String(n, m))**.** *Let n, m be such that $m < 2^{n/2}$. Alice receives m strings $x_1, x_2, \ldots, x_m \in \{0,1\}^n$ and Bob receives $y_1, y_2, \ldots, y_m \in \{0,1\}^n$. Their goal is to communicate with each other and output an n-bit string s such that s is not equal to $x_i \oplus y_j$ for every $1 \le i, j \le m$. Here $\oplus$ denotes the bit-wise XOR operation over strings.*

[3] In fact, lower bounds for Missing-String in the *algebraic query model* suffices. However, we find the algebraic query model somewhat counter-intuitive to work with (for example, the input to the query algorithms consists of a Missing-String instance along with its *low-degree extension*). The "transfer principle" [3, Section 4.3] allows us to simulate such query algorithms by communication protocols, hence we choose to study communication complexity.

[4] Suppose that $m \le 2^n/10$, Alice has inputs $x_1, x_2, \ldots, x_m \in \{0,1\}^n$ and Bob has inputs $y_1, y_2, \ldots, y_m \in \{0,1\}^n$. For each string $s \in \{0,1\}^n$, it costs only $O(\log m)$ bits of communication to obtain the value $f(s) := |\{i : x_i \le s\}| + |\{i : y_i \le s\}|$. Hence, we can use binary search to find two (lexicographically) adjacent strings $\mathrm{pred}(s)$ and s such that $f(\mathrm{pred}(s)) = f(s)$ by communicating $O(n \log m)$ bits. Clearly, s is not in the set $\{x_1, \ldots, x_m, y_1, \ldots, y_m\}$.

We show that XOR-Missing-String is hard for various communication complexity models, and transfer our communication complexity lower bounds to algebrization barriers. We now discuss our results in more detail.

1.1 Barriers to pr-PostBPE Circuit Lower Bounds

Our first result is an algebrization barrier to proving pr-$\mathsf{PostBPE} \not\subseteq$ i. o.-$\mathsf{SIZE}[O(n)]$, i.e., an almost-everywhere circuit lower bound for pr-PostBPE. Here, pr-PostBPE is the class of promise problems decided by a probabilistic exponential-time machine with *postselection* [23]: conditioned on some event (which may happen with exponentially small probability), the machine outputs the correct answer with high probability. Postselection is a powerful computational resource: PostBPP contains MA (thus NP) [23] and PostBQP (polynomial-time *quantum* computation with postselection) equals PP [1].[5]

▶ **Theorem 3.** *There exists an oracle A_1 and its multilinear extension $\widetilde{A_1}$ such that*

$$\text{pr-}\mathsf{PostBPE}^{\widetilde{A_1}} \subseteq \text{i. o.-}\mathsf{SIZE}^{A_1}[O(n)].$$

In particular, this also implies

$$\text{pr-}\mathsf{MA}_{\mathsf{E}}^{\widetilde{A_1}} \subseteq \text{i. o.-}\mathsf{SIZE}^{A_1}[O(n)].$$

The proof of pr-$\mathsf{MA}_{\mathsf{E}} \not\subseteq \mathsf{P}/_{\text{poly}}$ is algebrizing [12, 39, 3]. Our result implies that improving this circuit lower bound to almost-everywhere requires non-algebrizing techniques.

Theorem 3 follows from a PostBPP communication lower bound for XOR-Missing-String:

▶ **Theorem 4.** *Let $n \geq 1$ and $20n \leq m < 2^{n/2}$ be integers. Any* PostBPP *communication protocol that solves* XOR-Missing-String(n, m) *with error $\leq 2^{-5n}$ must have communication complexity $\Omega(m)$.*

Assume that $n \ll m \ll 2^{o(n)}$. A trivial protocol for XOR-Missing-String is to output a uniformly random n-bit string, which has communication complexity $O(n)$ and error $m^2/2^n$. Thus, Theorem 4 states that even if postselection is allowed, if we want to beat the error bound of this trivial protocol ($\approx 2^{-n}$), then the amount of communication needed is close to the maximum ($\Omega(m)$).

Moreover, since the error of any *pseudodeterministic*[6] communication protocol can be reduced by a 2^{-k} factor by repeating the protocol $\Theta(k)$ times and taking the majority answer, Theorem 4 implies that any pseudodeterministic PostBPP protocol that solves XOR-Missing-String(n, m) (with correct probability, say, $2/3$) must have communication complexity $\Omega(m/n)$. This stands in stark contrast to the case without pseudodeterministic constraints, as the trivial protocol is correct with probability $1 - m^2/2^n \gg 2/3$. We also remark that lower bounds against pseudodeterministic PostBPP communication protocols

[5] For example, a PostBPP machine can solve NP-complete problems as follows. With exponentially small probability, output "No" and halt. If this does not happen, choose an NP witness uniformly at random and "kill yourself" if this witness is invalid (that is, we *postselect* on the event that our witness is valid). If the algorithm survives, it outputs "Yes." Conditioned on survival, with high probability, our algorithm outputs "Yes" on Yes instances and outputs "No" on No instances.

[6] A randomized protocol for a search problem is *pseudodeterministic* [20] if there is a *canonical* output that is correct and outputted with probability $\geq 2/3$. The trivial protocol for XOR-Missing-String that outputs a random guess is not pseudodeterministic, since running it two times using different randomness is likely to yield different answers.

suffice for constructing an oracle A such that $\mathsf{PostBPE}^{\widetilde{A}}$ has small A-oracle circuits; however, we will need the full power of Theorem 4 against every low-error protocol to prove that *the promise version* of $\mathsf{PostBPE}^{\widetilde{A}}$ has small A-oracle circuits.

1.2 Barriers to BPE Circuit Lower Bounds

Our second result is an algebrization barrier to proving $\mathsf{BPE} \not\subseteq \mathsf{SIZE}[O(n)]$, i.e., an *infinitely-often* circuit lower bound for BPE:

▶ **Theorem 5.** *There exists an oracle A_2 and its multilinear extension $\widetilde{A_2}$ such that*

$$\mathsf{BPE}^{\widetilde{A_2}} \subseteq \mathsf{SIZE}^{A_2}[O(n)].$$

Previously, Aaronson and Wigderson [3] constructed an oracle A and its *multiquadratic* extension $\widetilde{A}$ such that $\mathsf{BPEXP}^{\widetilde{A}} \subseteq \mathsf{P}^{A}/_{\mathrm{poly}}$. Their proof works with the algebraic query model directly, and it is unclear how to prove the same result for multilinear extensions using their techniques.

Besides demonstrating the power of our communication- and duality-based approach, an algebrization barrier w.r.t. multilinear extension also aligns better with the notion of *affine relativization* [7]. A statement $\mathcal{C} \subseteq \mathcal{D}$ is said to *affine relativize* if $\mathcal{C}^{\widetilde{A}} \subseteq \mathcal{D}^{\widetilde{A}}$ for every $\widetilde{A}$ that is the multilinear extension of some oracle A. The crucial difference from algebrization (in the sense of [3]) is that the left-hand side is also required to relativize with the multilinear extension. Affine relativization is arguably a cleaner and more robust notion than algebrization.[7] It is important to only take multilinear extensions, as [7] was only able to show that, e.g., $\mathsf{PSPACE}^{\widetilde{A}} \subseteq \mathsf{IP}^{\widetilde{A}}$ for every multilinear oracle $\widetilde{A}$; it is unclear if such a statement holds when $\widetilde{A}$ is merely multiquadratic. (This is also why Aaronson and Wigderson [3] choose to relativize IP with the low-degree extension of A but to relativize PSPACE with A itself.) In this regard, Theorem 5 also shows that $\mathsf{BPE} \not\subseteq \mathsf{P}/_{\mathrm{poly}}$ is not affine relativizing (since $\mathsf{BPE}^{\widetilde{A}} \subseteq \mathsf{SIZE}^{A}[O(n)] \subseteq \mathsf{SIZE}^{\widetilde{A}}[O(n)]$).

To obtain Theorem 5, we prove a lower bound for XOR-Missing-String *against multiple pseudodeterministic* BPP *protocols simultaneously* in the following model.

- There are t communication protocols $Q_1, Q_2, \ldots, Q_t$ and t inputs (X_1, Y_1), (X_2, Y_2), $\ldots$, (X_t, Y_t).
- The goal of protocol Q_i is to solve the input (X_i, Y_i). However, each protocol is given the inputs to other protocols as well. More precisely, in each Q_i, Alice gets $(X_1, X_2, \ldots, X_t)$ as inputs and Bob gets $(Y_1, Y_2, \ldots, Y_t)$ as inputs.[8]
- We say that the protocols ***succeed*** if *at least one* of the protocols Q_i outputs a correct answer.
- For proving algebrization barriers, each Q_i and (X_i, Y_i) should have a different size and correspond to different input lengths of the oracle, but we omit this detail in the informal description here. We refer the reader to the full version of our paper for details.

[7] For example, algebrization is not necessarily closed under *modus ponens*: Although it requires non-algebrizing techniques to prove, say, $\mathsf{NEXP} \not\subseteq \mathsf{P}/_{\mathrm{poly}}$, it might still be possible to exhibit a complexity class $\mathcal{C}$ and prove both $\mathcal{C} \subseteq \mathsf{NEXP}$ and $\mathcal{C} \not\subseteq \mathsf{P}/_{\mathrm{poly}}$ via algebrizing techniques. In contrast, affine relativization is clearly closed under *modus ponens*.

[8] We remark that in the case of XOR-Missing-String, each X_i and Y_i is already a list of strings, which means Alice and Bob get t lists of strings each.

The point of this model is that the protocols are allowed to look at other protocols' inputs and to perform *win-win analyses*: the correctness of Q_i may rely on the *failure* of some other protocol $Q_{i'}$.[9] Indeed, this scenario models a variety of win-win analyses in complexity theory; see e.g., [15, Section 1.4.2] and [34, Section 5.2]. Circuit lower bounds for MA [12, 39] can also be modeled as win-win algebraic query algorithms for Missing-String; see Appendix A of the full version.

We show that efficient pseudodeterministic protocols require large communication complexity to solve XOR-Missing-String, even if they receive multiple instances and are allowed to perform win-win analyses in the above sense.

▶ **Theorem 6** (Informal). *In the above model, suppose that each (X_i, Y_i) is an instance of* XOR-Missing-String(n_i, m_i) *but the communication complexity of each Q_i is "much less" than m_i. Then there exists a sequence of inputs $\{(X_i, Y_i)\}_{i\in[t]}$ such that every Q_i fails to solve its corresponding instance (X_i, Y_i) pseudodeterministically.*

1.3 Barriers to $\mathsf{MA_E}$ Circuit Lower Bounds

Finally, we present algebrization barriers to improving the half-exponential lower bound for $\mathsf{MA_E}$ [12]. While we are unable to fully resolve the algebrizing circuit complexity of $\mathsf{MA_E}$, we show that for a natural subclass of $\mathsf{MA_E}$ which includes the hard language in [12], non-algebrizing techniques are required to go beyond half-exponential bounds.

Recall that a standard algorithm $M^{\widetilde{A}}$ in $\mathsf{MA_E^{\widetilde{A}}}$ defines a hard language if, *relative to this particular oracle* $\widetilde{A}$, $M^{\widetilde{A}}$ satisfies the MA promise and defines a language without small A-oracle circuits; we do not care about the behavior of $M^{\widetilde{B}}$ for other oracles $\widetilde{B}$. In contrast, we say that an $\mathsf{MA_E}$ machine M is a ***robust*** machine for defining a hard language (or simply "is robust"), if for *any* oracle B and its multilinear extension $\widetilde{B}$ such that $M^{\widetilde{B}}$ satisfies the MA promise, the language computed by $M^{\widetilde{B}}$ does not have small B-oracle circuits. The use of interactive proofs in [12] naturally leads to hard languages defined by robust MA machines; see appendix A of the full version for details.[10]

We use Rob-MA^A (resp. Rob-$\mathsf{MA_E^A}$) to denote the class of languages computed by robust MA (resp. $\mathsf{MA_E}$) algorithms with access to oracle A. Our main result is that proving a super-half-exponential bound for languages in E or robust $\mathsf{MA_E}$ would require non-algebrizing techniques:

▶ **Theorem 7** (Informal). *There exists an oracle A_3 and its multilinear extension $\widetilde{A_3}$ such that both $\mathsf{E}^{\widetilde{A_3}}$ and* Rob-$\mathsf{MA_E^{\widetilde{A_3}}}$ *admit half-exponential size A_3-oracle circuits.*

---

[9] If the possibility of such win-win analyses sounds surprising, it may help to compare it with the following classic puzzle. There are N prisoners, each assigned a (not necessarily distinct) number between 0 and $N-1$. Each prisoner can see others' numbers but not their own. Each prisoner must guess their own number simultaneously. If at least one prisoner guesses correctly, they are all set free; if none of them do, they are all executed. There is a simple solution that guarantees the prisoners' freedom. The i-th prisoner ($0 \le i \le N-1$) assumes that the total sum of all numbers is congruent to i modulo N, and then infers their own number as (i − sum of other prisoners' numbers) mod N. Exactly one assumption will be correct, so that prisoner will guess their own number correctly.

[10] In fact, the definition of robust MA resembles closer to the class $\mathsf{MA} \cap \mathsf{coMA}$ compared to MA itself. This is natural since *single-valued* FMA*-constructions of hard truth tables* (see, e.g., [15, Section 1.2.1]) give rise to circuit lower bounds for $\mathsf{MA} \cap \mathsf{coMA}$, and indeed the lower bound proved in [12] is for hard languages in $\mathsf{MA_E} \cap \mathsf{coMA_E}$. For ease of notation we use "robust MA" (Rob-MA) instead of "robust $\mathsf{MA} \cap \mathsf{coMA}$" (Rob-$\mathsf{MA} \cap \mathsf{coMA}$).

Why study robust $\mathsf{MA_E}$? We present two reasons for studying the notion of robust $\mathsf{MA_E}$.

Firstly, Theorem 7 is tight in the sense that, in the two cases of the win-win argument in [12], the hard language is either in E (i.e., deterministic exponential time), or in robust $\mathsf{MA_E}$. Therefore, a slight adaptation of [12] proves that for every oracle A and its multilinear extension $\widetilde{A}$, $\mathsf{E}^{\widetilde{A}} \cup \text{Rob-}\mathsf{MA}_{\mathsf{E}}^{\widetilde{A}}$ does not have sub-half-exponential size A-oracle circuits.

Secondly, while the classes E and $\text{Rob}_{h(n)}\text{-}\mathsf{MA_E}$ may initially appear arbitrary, they in fact capture two fundamental types of failures for algorithms in $\mathsf{MA_E} \cap \mathsf{coMA_E}$:

- For $M \in \text{Rob}_{h(n)}\text{-}\mathsf{MA_E}$, if $M^{\widetilde{A}}$ fails to achieve high circuit complexity, it must be due to an input x on which $M^{\widetilde{A}}$ is semantically incorrect. That is, the corresponding truth table entry is undefined. We call this a failure due to **no positive**.
- In contrast, for $M \in \mathsf{E}$, the value $M^{\widetilde{A}}(x)$ is always well-defined for all x, regardless of the oracle A. If $M^{\widetilde{A}}$ has low circuit complexity, it must be because its truth table is easy for A-oracle circuits. We call this a failure due to a **false positive**.

In general, for an algorithm $M \in \mathsf{MA_E} \cap \mathsf{coMA_E}$, failure to achieve high circuit complexity can be attributed to both types, depending on the oracle. Thus, E and $\text{Rob}_{h(n)}\text{-}\mathsf{MA_E}$ represent two extreme cases within $\mathsf{MA_E} \cap \mathsf{coMA_E}$.

Our proof of Theorem 7 entails techniques that can force the two types of failures to occur. In this sense, the proof says something fundamental about $\mathsf{MA_E} \cap \mathsf{coMA_E}$. However, our current methods are limited to handling algorithms that exhibit only a single failure mode. For more general algorithms in $\mathsf{MA_E} \cap \mathsf{coMA_E}$ that can have mixed types of failures, our understanding remains incomplete.

Proof of Theorem 7 via a communication lower bound. To prove algebrization barriers to circuit lower bounds for robust $\mathsf{MA_E}$, we need to understand the *robust* MA communication complexity of XOR-Missing-String. Here, an MA protocol P is called *robust* if on every input (X, Y), Merlin (i.e., the prover) cannot convince the protocol to output a non-solution for (X, Y) except with very small probability. Underlying Theorem 7 is the following lower bound stating that efficient MA protocols for XOR-Missing-String that are robust can only be correct on a tiny fraction of inputs:

▶ **Lemma 8.** *Let $n \geq 1, 1 \leq m < 2^{n/2}$ be parameters. Let P be a robust* FMA *communication protocol of complexity C attempting to solve* XOR-Missing-String(n, m). *Then the fraction of inputs that P solves is at most*

$$2^{-\Omega(m/C)+O(C+n)}.$$

Intuitively, since there is no efficient way to verify whether a string is a solution, any protocol that solves a large number of instances of XOR-Missing-String is essentially guessing the answer (similar to the naïve protocol that succeeds with probability $1 - 2^{-O(n)}$), which means that it must make mistakes. Since robust protocols are not allowed to output false positives, they can only be correct on a tiny fraction of inputs.

Although Lemma 8 is useful for analyzing the behaviors of robust MA protocols, it does not directly imply any *half-exponential* bounds. Indeed, we need to carefully diagonalize against both $\mathsf{E}^{\widetilde{A}}$ and $\text{Rob}_{h(n)}\text{-}\mathsf{MA}_{\mathsf{E}}^{\widetilde{A}}$ simultaneously, which leads to a half-exponential bound. We discuss this in more detail in Subsection 1.4.

1.4 Technical Overview

Next, we briefly overview our proof techniques. In fact, the engine behind all of our communication lower bounds is the following observation about XOR-Missing-String:

▶ **Lemma 9** (Informal version of Lemma 17). *For every large enough rectangle $R = \mathcal{X} \times \mathcal{Y}$ and every answer s, there is a large enough subrectangle $R' = \mathcal{X}' \times \mathcal{Y}'$ ($\mathcal{X}' \subseteq \mathcal{X}$ and $\mathcal{Y}' \subseteq \mathcal{Y}$) such that s is a* wrong answer *for the* XOR-Missing-String *problem on every input $(X, Y) \in R'$.*

With this lemma in hand, it is not hard to prove Theorem 4. We characterize a PostBPP communication protocol as a distribution over labeled rectangles (using *approximate majority covers* [29]), where for each input (X, Y), the output of the protocol is given by the label of a randomly selected rectangle that contains (X, Y). Using Lemma 9, for every large enough labeled rectangle, there exists a large subrectangle in which the label is never a solution to XOR-Missing-String. Lemma 9 thus translates to a lower bound for the success probability of large labeled rectangles (i.e., the probability that the label is a solution, over a random input in the rectangle). This then translates to a lower bound for (weighted) average success probability over a random input, because the distribution consists mostly of large rectangles when the protocol is efficient. This lower bound implies that efficient protocols cannot have very high success probability.

However, more work needs to be done to obtain our other communication lower bounds and algebrization barriers.

BPP communication lower bounds. We prove our pseudodeterministic BPP communication lower bound in the "win-win model" (Theorem 6) via a reduction to the PostBPP communication lower bound (Theorem 4). We first use an induction on the number of protocols; assume towards a contradiction that Theorem 6 holds for any sequence of $t-1$ protocols but does not hold for some sequence of t protocols $Q_1, \ldots, Q_t$. Then the following PostBPP communication protocol solves XOR-Missing-String efficiently: Given an input (X, Y), guess $t-1$ inputs $\{(X_i, Y_i)\}_{i \in [t-1]}$ uniformly at random, set $(X_t, Y_t) := (X, Y)$, and *postselect* on the event that all of the first $t-1$ protocols fail. That is, for every $1 \leq i \leq t-1$, Q_i fails to solve (X_i, Y_i) given $\{(X_i, Y_i)\}_{i \in [t]}$. By our induction hypothesis, this event happens with probability strictly greater than 0. If this happens, then Q_t solves (X_t, Y_t) correctly. This contradicts Theorem 4.

For clarity, we have made a crucial simplification in the above description: To obtain algebrization barriers, we also need to prove lower bounds against *list-solvers* for XOR-Missing-String, which are communication protocols that output a *small set* of answers such that one of the answers is correct. Fortunately, it is not hard to adapt the proof of Theorem 4 to prove a PostBPP communication lower bound for list solvers; see the full version for details.

MA communication lower bounds against robust protocols. A robust protocol can be described as a collection of randomized verifiers $V_{w,\pi}$, where $V_{w,\pi}(X, Y) = 1$ means that when the input is (X, Y) and π is given as proof, the protocol accepts w as output. In the formal proof, we apply error reduction on the verifiers, so that when w is not a solution to (X, Y), $V_{w,\pi}(X, Y) = 1$ with extremely small probability.

Thus, if a protocol has large success probability, there must exist some answer string w and some proof π, such that over a random input (X, Y), $\Pr[V_{w,\pi}(X, Y) = 1]$ is large. We interpret this $V_{w,\pi}$ as a distribution over labeled rectangles, and apply Lemma 9 to show that conditioned on $V_{w,\pi}(X, Y) = 1$, the verifier makes many mistakes (i.e., $\Pr[V_{w,\pi}(X, Y) = 1 \wedge w$ is not a solution to $(X, Y)]$ is large). This implies that the protocol must make mistakes on some (X, Y), so it cannot be robust.

The half-exponential barrier for robust $\mathsf{MA_E}$. We formulate the problem in terms of refuting communication protocols, where we need to refute a finite set of protocols (deterministic or robust) on each input length. More formally, to construct an algebrized world with circuit upper bound $h(n)$, our oracle encodes one instance (X_i, Y_i) of XOR-Missing-String$(2^i, 2^{h(i)})$ for each input length i. Each protocol being diagonalized is dedicated to solving the one instance that corresponds to its input length (it can nevertheless see the other instances).

We employ an inductive approach to construct the oracle. At step i, we maintain a rectangle[11] R_i of oracles, such that the rectangle is relatively large, and any protocol that works on input length at most i fails to solve its instance on any oracle $A \in R_i$.

The goal of every step is then to slightly shrink the rectangle R_{i-1} into R_i, so that the protocols working on length i are refuted. To achieve this, we employ a combination of two different strategies:

- **Refuting deterministic protocols:** Given a deterministic protocol P working on length i and a large enough rectangle R_i, we can always find a large subrectangle $R' \subseteq R_i$ such that P outputs the same answer on every oracle in R' (this is by definition of deterministic protocols). By Lemma 9, there exists a large subrectangle $R'' \subseteq R'$, such that for any oracle $A \in R''$, P does not solve (X_i, Y_i) on A. We then update $R_i \leftarrow R''$ to refute P.
- **Refuting robust protocols:** Given a robust protocol Q working on length i and a large enough rectangle R_i, a random oracle from R_i refutes Q with high probability. This is true as long as the density of R_i is sufficiently larger than the success probability of Q on a uniformly random input (as upper bounded in Lemma 8).

On their own, both strategies work extremely well: The first strategy is very similar to the proof of $\mathsf{NEXP}^{\widetilde{A}} \subset \mathsf{P}^A/\mathrm{poly}$ by [3], and can be used to obtain much better circuit upper bounds than half-exponential. For the second strategy, since robust protocols are very weak, a randomly selected oracle would refute all robust protocols with high probability.

However, it is unclear how to combine the two strategies. On the one hand, after refuting a deterministic protocol P by the first strategy, the resulting rectangle R_i may be too small for the second strategy to work. On the other hand, although the set of oracles $R' \subseteq R_i$ that refutes a robust protocol Q is large, it may not be a rectangle, so we cannot use it to refute the next deterministic protocol.

To integrate these two strategies, we refute each robust protocol at a carefully chosen time step. Suppose that we refute a robust protocol Q_i working on length i *after* all deterministic protocols working on lengths $< k$ are refuted, where $k = k(i)$ is some function on i. Also, recall that we want to prove an algebrized circuit upper bound of size $h(n)$.

- After refuting the deterministic protocols working on lengths $< k$, we end up with a rectangle that contains roughly a $2^{-2^{O(k)}}$ fraction of all oracles. This is because the deterministic protocols working on lengths $< k$ run in time $2^{O(k)}$. In contrast, Lemma 8 implies that Q_i only solves a $\approx 2^{-2^{h(i)}}$ fraction of oracles (since it attempts to solve an instance of XOR-Missing-String$(2^i, 2^{h(i)})$). Therefore, if $2^{-2^{O(k)}} \gg 2^{-2^{h(i)}}$, then we can refute Q_i.
- On the other hand, after refuting Q_i, we still need to refute the deterministic protocols working on length k. Since these protocols attempt to solve XOR-Missing-String$(2^k, 2^{h(k)})$, to apply Lemma 9, this requires our current rectangle R_k to have size

[11] We interpret our oracles as the concatenation of Alice's and Bob's inputs, so it is reasonable to talk about a "rectangle" of oracles.

at least roughly $2^{-2^{h(k)}}$. Since Q_i is equivalent to a deterministic protocol running in time $2^{2^{O(i)}}$ (by enumerating the random bits and Merlin's proof), after refuting it, we still have a rectangle of density $\approx 2^{-2^{2^{O(i)}}}$. We thus need that $2^{-2^{2^{O(i)}}} \gg 2^{-2^{h(k)}}$.

Overall, we require that $k \ll h(i)$ and $2^i \ll h(k)$. This implies that h must be super-half-exponential.

1.5 Related Works

Prior work on super-polynomial circuit lower bounds. Kannan's seminal work [27] proved a super-polynomial size lower bound for $\Sigma_2\mathsf{EXP}$; since then, a sequence of work has proved circuit lower bounds for various complexity classes such as $\mathsf{ZPEXP}^{\mathsf{NP}}$ [31, 11], $\mathsf{S}_2\mathsf{EXP}$ [14, 13], PEXP [42, 2], $\mathsf{MA}_{\mathsf{EXP}}$ [12, 39], $\mathsf{ZPEXP}^{\mathrm{MCSP}}$ [26, 24], and more [41, 16]. Such lower bounds are usually proved by *Karp–Lipton* theorems [28, 9, 18]: If a large uniform class (usually PH or EXP) admits polynomial-size circuits, then it collapses to a smaller uniform class (such as $\Sigma_2\mathsf{P}$). As explained in [36] and [15, Section 1.4.1], such a strategy naturally yields a *half-exponential* bound. Recently, [15, 33] proved near-maximum ($2^n/n$) circuit lower bounds for the classes $\Sigma_2\mathsf{EXP}$, $\mathsf{ZPEXP}^{\mathsf{NP}}$, and $\mathsf{S}_2\mathsf{EXP}$, improving upon previous half-exponential bounds. Near-maximum circuit lower bounds are also known for several classes with *subexponential* amount of advice bits, such as $\mathsf{BPEXP}^{\mathrm{MCSP}}/_{2^{n^\varepsilon}}$ [24] and $\mathsf{AMEXP}/_{2^{n^\varepsilon}}$ [16].

Algebrization. The interactive proof results such as $\mathsf{IP} = \mathsf{PSPACE}$ [35, 40] and $\mathsf{MIP} = \mathsf{NEXP}$ [8] generated much excitement among complexity theorists, as they are the first "truly compelling" ([4]) non-relativizing results in complexity theory. There has been much discussion on the extent to which these results are non-relativizing, and how to adapt the relativization barrier to accommodate these results [6, 19]. Arguably, the most influential work in this direction is that of Aaronson and Wigderson [3] on the algebrization barrier, which nicely captures interactive proof results and arithmetization-based techniques. However, Aaronson and Wigderson's algebrization barrier has its own subtleties (such as not being closed under *modus ponens*, see footnote 7), which leads to several proposed refinements of its definition [25, 7]. We also mention the *bounded relativization* barrier, recently proposed by Hirahara, Lu, and Ren [24], which attempts to capture the interactive proof results entirely within the original Baker–Gill–Solovay [10] framework of relativization.

1.6 Open Problems

We think the most important open problem left in our paper is to study the algebrizing circuit complexity of MA_{E}. Can we strengthen Theorem 7 to hold for all of MA_{E} instead of just "robust" MA_{E} algorithms? If not, can we prove an exponential size lower bound for MA_{E} (potentially with one bit of advice) using algebrizing techniques?

Another open question is to prove a PP communication lower bound for XOR-Missing-String, which would imply an algebrization barrier to proving almost-everywhere circuit lower bounds for PEXP. Such circuit lower bounds are known in the infinitely-often regime [12, 42].

Finally, our work did not examine the regime of fixed-polynomial circuit lower bounds. Using algebrizing techniques, Santhanam [39] proved that for every constant $k \geq 1$, $\mathsf{MA}/_1 \not\subseteq \mathsf{SIZE}[n^k]$. Aaronson and Wigderson [3] speculated that it might be possible to eliminate the advice bit and prove $\mathsf{MA} \not\subseteq \mathsf{SIZE}[n^k]$ using "tried-and-true arithmetization methods," but there has been no progress in this direction. Is there an algebrizing barrier to proving $\mathsf{MA} \not\subseteq \mathsf{SIZE}[n^k]$?

2 Preliminaries

▶ **Definition 10.** *A* **search problem** f *over domain* $\mathcal{X} \times \mathcal{Y}$ *and range* $\mathcal{O}$ *is defined using a relation* $\mathcal{R} \subset (\mathcal{X} \times \mathcal{Y}) \times \mathcal{O}$*. For any input* $(X, Y) \in \mathcal{X} \times \mathcal{Y}$*, the valid solutions for* f *on* (X, Y) *are those* s *such that* $(X, Y, s) \in \mathcal{R}$ *(we say that* $\boldsymbol{s}$ **is a solution to** $\boldsymbol{f(X, Y)}$*). We say that* f *is a* **total problem***, if for any* (X, Y)*, there is at least one valid solution.*

2.1 Complexity Classes

We assume familiarity with basic complexity classes such as P, NP, BPP and their exponential-time versions EXP, NEXP, BPEXP. The reader is encouraged to consult standard textbooks [5, 21] or the Complexity Zoo[12] for their definition.

▶ **Definition 11** (PostBPP)**.** *A promise problem* $\Pi = (\Pi_{\text{yes}}, \Pi_{\text{no}})$ *is in* pr-PostBPP *if there exist two polynomial-time algorithms* A, B*, taking an input* $x \in \{0,1\}^n$ *and randomness* $r \in \{0,1\}^{\text{poly}(n)}$ *(they take the* same *randomness), such that the following holds for every* $x \in \Pi_{\text{yes}} \cup \Pi_{\text{no}}$*.*

- *If* $x \in \Pi_{\text{yes}}$*, then* $\Pr_r[A(x, r) = 1 \mid B(x, r) = 1] \geq 2/3$*.*
- *If* $x \in \Pi_{\text{no}}$*, then* $\Pr_r[A(x, r) = 1 \mid B(x, r) = 1] \leq 1/3$*.*
- $\Pr_r[B(x, r) = 1] > 0$*.*

We say that the algorithm **postselects** *on the event that* $B(x, r) = 1$*.*

▶ **Definition 12** (MA ∩ coMA)**.** *A promise problem* $\Pi = (\Pi_{\text{yes}}, \Pi_{\text{no}})$ *is in* pr-(MA ∩ coMA) *if there exist two polynomial-time algorithms (verifiers)* V_0, V_1*, taking an input* $x \in \{0,1\}^n$*, a proof* $\pi \in \{0,1\}^{\text{poly}(n)}$ *and randomness* $r \in \{0,1\}^{\text{poly}(n)}$ *(they take* different *randomness), such that the following holds for every* $x \in \Pi_{\text{yes}} \cup \Pi_{\text{no}}$*.*

- *If* $x \in \Pi_{\text{yes}}$*, then* V_1 *accepts* x *and* V_0 *rejects* x*.*
- *If* $x \in \Pi_{\text{no}}$*, then* V_0 *accepts* x *and* V_1 *rejects* x*.*

Here, for $k = 0, 1$*, we say that*

- V_k **accepts** x*, if there exists a proof* π *such that* $\Pr_r[V_k(x, \pi, r) = 1] \geq 2/3$*.*
- V_k **rejects** x*, if for any proof* π*, we have that* $\Pr_r[V_k(x, \pi, r) = 1] \leq 1/3$*.*

2.2 Algebrization

In this section, we present some definitions regarding algebrization barriers. The definitions are in accordance with [3], with slight modifications.

▶ **Definition 13** (Oracle)**.** *An* **oracle** A *is a collection of Boolean functions* $A_m : \{0,1\}^m \to \{0,1\}$*, one for each* $m \in \mathbb{N}$*. Given a complexity class* $\mathcal{C}$*, by* $\mathcal{C}^A$ *we mean the class of languages decidable by a* $\mathcal{C}$ *machine that can query* A_m *for any* m *of its choice.*

We say that a separation $\mathcal{C} \not\subset \mathcal{D}$ does not algebrize, if there exist an oracle A and a low-degree extension $\widetilde{A}$ of A, such that $\mathcal{C}^{\widetilde{A}} \subset \mathcal{D}^A$. Throughout this paper, we only consider multilinear extensions, which are the simplest class of low-degree extensions.

▶ **Definition 14** (Multilinear Extension over Finite Fields)**.** *Let* $A_m : \{0,1\}^m \to \{0,1\}$ *be a Boolean function, and let* $\mathbb{F}$ *be a finite field. The (unique) multilinear extension of* A_m *over* $\mathbb{F}$ *is the linear function* $\widetilde{A}_{m,\mathbb{F}} : \mathbb{F}^m \to \mathbb{F}$ *that agrees with* A_m *on* $\{0,1\}^m$*. Given an oracle* $A = (A_m)$*, the* **multilinear extension** $\tilde{A}$ *of* A *is the collection of multilinear extensions* $\widetilde{A}_{m,\mathbb{F}}$*, one for each positive integer* m *and finite field* $\mathbb{F}$*.*

[12] `https://complexityzoo.net/`, accessed Nov 28, 2025.

Given a complexity class $\mathcal{C}$, by $\mathcal{C}^{\widetilde{A}}$ we mean the class of languages decidable by a $\mathcal{C}$ machine that can query $\widetilde{A}_{m,\mathbb{F}}$ for any $m, \mathbb{F}$ of its choice. Moreover, we assume that each query to $\widetilde{A}_{m,\mathbb{F}}$ takes $O(m \cdot \log |\mathbb{F}|)$ time.

An important property of multilinear extensions is that algorithms having access to $\widetilde{A}$ can be simulated by communication protocols where Alice and Bob are each given half of A as input. This reduces proving algebrization barriers to proving communication lower bounds. Formally, we have the following theorem, which is a simple corollary of [3, Theorem 4.11].

▶ **Theorem 15.** *Let A be an oracle, and let A_0 (resp. A_1) be the subfunction of A obtained by restricting the first bit to 0 (resp. 1). Let M be a deterministic algorithm that has oracle access to $\widetilde{A}$ and runs in time $T(|x|)$ when given x as input. There exists a deterministic communication protocol P in which Alice (resp. Bob) is given x and the function A_0 (resp. A_1) as input, such that P uses $O(T(|x|)^3)$ bits of communication, and agrees with $M(x)$ on any input x and any oracle A.*

We remark that the proof of [3, Theorem 4.11] can be generalized to produce randomized communication protocols from randomized algorithms, MA communication protocols from MA algorithms, and so on.

3 An Infinitely-Often Algebrization Barrier for PostBPE

In this section, we prove the following algebrization barrier:

▶ **Theorem 3.** *There exists an oracle A and its multilinear extension $\widetilde{A}$ such that*

$$\text{pr-}\mathsf{PostBPE}^{\widetilde{A}} \subseteq \text{i.o.-}\mathsf{SIZE}^{A}[O(n)].$$

3.1 PostBPP Communication Lower Bounds for XOR-Missing-String

Theorem 3 follows from the following communication lower bound for XOR-Missing-String:

▶ **Theorem 4.** *Let $n \geq 1$ and $20n \leq m < 2^{n/2}$ be integers. Any PostBPP communication protocol that solves XOR-Missing-String(n, m) with error $\leq 2^{-5n}$ must have communication complexity $\Omega(m)$.*

In this paper, PostBPP communication protocols are defined as follows:[13]

▶ **Definition 16** (PostBPP communication protocols for search problems)**.** *Let f be a search problem over domain $\mathcal{X} \times \mathcal{Y}$ and range $\mathcal{O}$. A* **PostBPP communication protocol** *Π for f is defined as follows: Let k be the length of the public randomness used by Π, and let $\{\Pi_r\}_{r \in \{0,1\}^k}$ be a set of deterministic communication protocols, each having communication complexity c. On input $(X, Y) \in \mathcal{X} \times \mathcal{Y}$, protocol Π first samples a uniformly random string $r \in \{0,1\}^k$, then simulates Π_r, whose output can be $\perp$ or any value from $\mathcal{O}$. The communication complexity of Π is defined to be $k + c$.*

We say that Π **solves f with error ϵ***, if for any input $(X, Y) \in \mathcal{X} \times \mathcal{Y}$,*

$$\Pr_r\left[\Pi_r(X,Y) \text{ is a solution to } f(X,Y) \,\middle|\, \Pi_r(X,Y) \neq \perp\right] \geq 1 - \epsilon.$$

[13] There are many different definitions in the literature (see e.g., [22]). Here, we choose a simple definition that is best suited for proving the barrier.

We say that Π *is* **pseudodeterministic**, *if for any input* $(X, Y) \in \mathcal{X} \times \mathcal{Y}$, *there exists some answer* $s \in \mathcal{O}$, *such that*

$$\Pr\left[\Pi_r(X,Y) = s \,\middle|\, \Pi_r(X,Y) \neq \perp\right] \geq 2/3.$$

The main technical observation for our lower bound is that for any large enough rectangle R, no single answer $s \in \{0,1\}^n$ will be correct for every input $(X, Y) \in R$. In fact, there is always a $2^{-\Theta(n)}$ fraction of inputs in R on which s is not a valid solution.

To use this result in the lower bound, note that a PostBPP communication protocol can be seen as a distribution of labeled rectangles (i.e., approximate majority covers, as defined in Subsubsection 3.1.2). If a protocol has small communication complexity, then it must contain many large rectangles, so we can apply the aforementioned lower bound on individual rectangles, which implies that the protocol must have error $2^{-O(n)}$.

3.1.1 A Lower Bound for One Rectangle

In this section, we prove the main technical lemma for the communication lower bound. It is helpful to draw an analogy from the decision tree version of Missing-String: Suppose that the input is a list of m n-bit strings. Whenever the decision tree only queries the input on $m-1$ bits, there always exists a string in the input list that is not queried. In this case, regardless of the decision tree's answer, the adversary can arbitrarily modify the value of that string and make the decision tree fail.

Here, we show that the XOR-Missing-String problem has a similar property: If, conditioned on the communication history, the rectangle formed by the possible inputs is large (this corresponds to the decision tree making a small number of queries), then regardless of the protocol's answer, there always exists a significant portion of the rectangle, on which the protocol fails.

▶ **Lemma 17.** *Let* $n \geq 1, 1 \leq m < 2^{n/2}$ *be integers. Let* $R \subseteq \{0,1\}^{nm} \times \{0,1\}^{nm}$ *be a rectangle (i.e.,* R *is of the form* $\mathcal{X} \times \mathcal{Y}$, *where* $\mathcal{X}, \mathcal{Y} \subseteq \{0,1\}^{nm}$*) of size at least* $2^{2nm-m+2}$. *Let* s *be any* n*-bit string. Then there exists some* R', *which is a subrectangle of* R *and has size at least* $2^{-2n-2}|R|$, *such that for any* $(X, Y) \in R'$, s *is not a solution to* XOR-Missing-String *on* (X, Y).

Proof. Without loss of generality, we only have to prove the lemma for the case where $s = 0^n$. When $s \neq 0^n$, we consider a different rectangle R_s, in which every input string x for Alice in every input list $X \in \mathcal{X}$ is replaced by $x \oplus s$. By the definition of XOR-Missing-String, the lemma holds for R and s if and only if it holds for R_s and 0^n.

To prove the lemma, we need to show that there exists a large enough subrectangle R' in R, in which 0^n is never a solution to XOR-Missing-String. Recall that R is equal to $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X}$ is a set of Alice's inputs and $\mathcal{Y}$ is a set of Bob's inputs. For any string $s \in \{0,1\}^n$, let $\mathcal{X}_s = \{X \in \mathcal{X}, s \in X\}$ denote the subset of $\mathcal{X}$ that contains the string s; $\mathcal{Y}_s$ is defined similarly.

For any $s \in \{0,1\}^n$, consider the subrectangle $\mathcal{X}_s \times \mathcal{Y}_s \subseteq R$. Since any input list (Alice's or Bob's) in the subrectangle always contains s, 0^n is never a solution in $\mathcal{X}_s \times \mathcal{Y}_s$.

In the remaining, we show that there exists some s such that $\mathcal{X}_s \times \mathcal{Y}_s$ is large enough (i.e., has size $\geq 2^{-2n-2}|R|$) using a counting argument. If we can show this, then the lemma follows by letting $R' = \mathcal{X}_s \times \mathcal{Y}_s$.

Let $cx_s = |\mathcal{X}_s|$ denote the number of occurrences of string s in $\mathcal{X}$. Note that, if some input $X \in \mathcal{X}$ contains multiple copies of s, X is only counted once in cx_s. Let $x_1, \ldots, x_{2^n}$ be all the n-bit strings, sorted in non-increasing order of cx. Similarly define cy_s and $y_1, \ldots, y_{2^n}$. We have the following lower bound on the number of occurrences of x_i:

▷ Claim 18. For any $1 \leq i \leq 2^n$, cx_{x_i} is at least

$$\frac{|\mathcal{X}| - (i-1)^m}{2^n - i + 1}.$$

Proof. Let $k = \sum_{i' \geq i} cx_{x_{i'}}$, i.e., the sum of occurrences of $x_{\geq i}$.

Consider the inputs X in $\mathcal{X}$ that only contain strings from $x_1, \ldots, x_{i-1}$. The number of such X is at most $(i-1)^m$. Therefore, at least $|\mathcal{X}| - (i-1)^m$ inputs in $\mathcal{X}$ contain at least one string from $x_{\geq i}$. Each of the $|\mathcal{X}| - (i-1)^m$ inputs contribute at least one to the sum k. That is,

$$k = \sum_{i' \geq i} cx_{x_{i'}} \geq |\mathcal{X}| - (i-1)^m.$$

Since x_i occurs at least as frequently as any string in $x_{i+1}, \ldots, x_{2^n}$, we have that $k \leq cx_{x_i} \cdot (2^n - i + 1)$. Therefore,

$$cx_{x_i} \geq \frac{k}{2^n - i + 1} \geq \frac{|\mathcal{X}| - (i-1)^m}{2^n - i + 1}.$$ ◁

The same bound also applies to cy_{y_i}. Fix any $1 \leq i \leq 2^n$ and consider the strings $x_1, \ldots, x_i$ and $y_1, \ldots, y_{2^n - i + 1}$. By the pigeonhole principle, there must exist some string s that appears in both $\{x_1, \ldots, x_i\}$ and $\{y_1, \ldots, y_{2^n - i + 1}\}$. Using Claim 18, we can show that $\mathcal{X}_s \times \mathcal{Y}_s$ is large, that is,

$$\begin{aligned}
|\mathcal{X}_s \times \mathcal{Y}_s| &= cx_s \cdot cy_s \\
&\geq cx_{x_i} \cdot cy_{y_{2^n - i + 1}} && \text{(by definition of the lists } x, y) \\
&\geq \frac{|\mathcal{X}| - (i-1)^m}{2^n - i + 1} \cdot \frac{|\mathcal{Y}| - (2^n - i)^m}{i}. && \text{(by Claim 18)}
\end{aligned}$$

Let $A = |\mathcal{X}|, B = |\mathcal{Y}|$. It now remains to show that, for any A, B such that $1 \leq A, B \leq 2^{nm}$ and $A \cdot B \geq 2^{2nm - m + 2}$, there exists some $1 \leq i \leq 2^n$ such that the rectangle chosen above is large enough. That is,

$$\frac{A - (i-1)^m}{2^n - i + 1} \cdot \frac{B - (2^n - i)^m}{i} \geq 2^{-2n-2} \cdot A \cdot B. \tag{1}$$

Let $i^* \geq 1$ be the largest integer such that $2(i^* - 1)^m \leq A$. Note that $2 \cdot 2^{nm} > A$, so we must have $i^* \leq 2^n$. In this case, we have that $2(2^n - i^*)^m \leq B$, since if not, then

$$\begin{aligned}
& 2(i^*)^m \cdot 2(2^n - i^*)^m > A \cdot B && ((i^*)^m > A \text{ by definition of } i^*) \\
\Rightarrow\ & (i^* \cdot (2^n - i^*))^m > A \cdot B/4 \geq 2^{2nm - m} \\
\Rightarrow\ & i^* \cdot (2^n - i^*) > 2^{2n-1},
\end{aligned}$$

which is impossible since $i \cdot (2^n - i) \leq 2^{2(n-1)}$ for any i. Therefore, by plugging i^* into (1), we have that

$$\begin{aligned}
& \frac{A - (i^* - 1)^m}{2^n - i^* + 1} \cdot \frac{B - (2^n - i^*)^m}{i^*} \\
\geq\ & \frac{A/2}{2^n - i + 1} \cdot \frac{B/2}{i^*} && (2(i^* - 1)^m \leq A \text{ and } 2(2^n - i^*)^m \leq B) \\
\geq\ & \frac{A/2}{2^n} \cdot \frac{B/2}{2^n} = 2^{-2n-2} \cdot A \cdot B.
\end{aligned}$$ ◀

3.1.2 Reducing PostBPP Communication to Approximate Majority Covers

To use Lemma 17 in the lower bound proof, we use the characterization of PostBPP communication protocols by distributions of labeled rectangles called *approximate majority covers* [29].

▶ **Definition 19** (approximate majority covers). *Let f be a search problem over the domain $\mathcal{X} \times \mathcal{Y}$ and range $\mathcal{O}$. An* **approximate majority cover for f** *is a (multi) set of labeled rectangles $\mathcal{AMC} = \{(R, s)\}$, where $R \subseteq \mathcal{X} \times \mathcal{Y}$ is a rectangle, i.e., R is of the form $A \times B$ for some $A \subseteq \mathcal{X}, B \subseteq \mathcal{Y}$, and $s \in \mathcal{O}$ is the label of R. The* **size** *of $\mathcal{AMC}$ is defined as the number of labeled rectangles.*

We say that the approximate majority cover **solves f with error ϵ**, *if for any $(X, Y) \in \mathcal{X} \times \mathcal{Y}$, we have that*

$$\Pr_{(R,s)\sim\mathcal{AMC}}[s \text{ is a solution to } f(X, Y) \mid (X, Y) \in R] \geq 1 - \epsilon.$$

That is, when we randomly sample from $\mathcal{AMC}$ a labeled rectangle (R, s) that contains (X, Y), the label s is a solution with probability $2/3$.

▷ Claim 20. Let f be a total search problem over domain $\mathcal{X} \times \mathcal{Y}$ and range $\mathcal{O}$. If there exists a PostBPP communication protocol Π that solves f with error ϵ and has complexity C, then there exists an approximate majority cover that solves f with error ϵ and has size $\leq 2^C$.

Proof. Assume that Π uses $k \leq C$ bits of randomness. That is, Π is the uniform distribution over 2^k deterministic communication protocols $\{\Pi_r\}$, where each protocol Π_r has complexity $C - k$.

Construct an approximate majority cover $\mathcal{AMC}$ as follows: Initially, $\mathcal{AMC}$ is empty. Each deterministic protocol Π_r partitions the input space into at most 2^{C-k} pairwise disjoint rectangles, where the answers of Π_r in each rectangle are the same. For each such rectangle R of Π_r, suppose the answer of Π_r on R is v, we add (R, v) to $\mathcal{AMC}$ if and only if $v \neq \perp$.

It is clear that the size of $\mathcal{AMC}$ is at most 2^C. Moreover, for any $(X, Y) \in \mathcal{X} \times \mathcal{Y}$, there is a one-to-one correspondence between the rectangles in $\mathcal{AMC}$ containing (X, Y) and the protocols Π_r for which $\Pi_r(X, Y) \neq \perp$. Therefore, the value of $\Pi(X, Y)$ is distributed the same as the label of a uniformly random rectangle in $\mathcal{AMC}$ that contains (X, Y). ◁

3.1.3 Putting Everything Together

With the components in place, we can now prove Theorem 4.

▶ **Theorem 4.** *Let $n \geq 1$ and $20n \leq m < 2^{n/2}$ be integers. Any* PostBPP *communication protocol that solves* XOR-Missing-String(n, m) *with error $\leq 2^{-5n}$ must have communication complexity $\Omega(m)$.*

Proof. Let Π be a PostBPP communication protocol that solves XOR-Missing-String(n, m) with error $\leq 2^{-5n}$, and let C denote the complexity of Π. We convert Π into an approximate majority cover $\mathcal{AMC}$ using Claim 20. We have that $\mathcal{AMC}$ solves XOR-Missing-String(n, m) with error $\leq 2^{-5n}$ and has size $2^{O(C)}$. We now show that the size of $\mathcal{AMC}$ is at least $2^{\Omega(m)}$, which proves the lemma.

Let $S = \sum_{(R,s)\in\mathcal{AMC}} |R|$ denote the total size of the rectangles. Since $\mathcal{AMC}$ is correct, each input $(X, Y) \in \{0, 1\}^{nm} \times \{0, 1\}^{nm}$ must be contained in at least one rectangle, which means that $S \geq 2^{2nm}$.

Let

$$C_{\text{correct}} \coloneqq \sum_{(X,Y)\in\{0,1\}^{nm}\times\{0,1\}^{nm}} \sum_{(R,s)\in\mathcal{AMC},R\ni(X,Y)} [s \text{ is a solution to XOR-Missing-String on } (X,Y)].$$

Since $\mathcal{AMC}$ solves XOR-Missing-String(n,m) with error 2^{-5n}, we have that

$$\begin{aligned}
&C_{\text{correct}} \\
&\geq \sum_{(X,Y)\in\{0,1\}^{nm}\times\{0,1\}^{nm}} (1-2^{-5n}) \cdot \sum_{(R,s)\in\mathcal{AMC},R\ni(X,Y)} 1 \quad \text{(each } (X,Y) \text{ is correct w.h.p.)} \\
&= (1-2^{-5n}) \cdot \sum_{(R,s)\in\mathcal{AMC}} \sum_{(X,Y)\in R} 1 \\
&= (1-2^{-5n}) \cdot S.
\end{aligned}$$

On the other hand,

$$\begin{aligned}
&C_{\text{correct}} \\
&= \sum_{(R,s)\in\mathcal{AMC}} \sum_{(X,Y)\in R} [s \text{ is a solution to XOR-Missing-String on } (X,Y)] \\
&= \sum_{\substack{(R,s)\in\mathcal{AMC} \\ |R|\geq 2^{2nm-m+2}}} \sum_{(X,Y)\in R} [s \text{ is a solution to XOR-Missing-String on } (X,Y)] \\
&\quad + \sum_{\substack{(R,s)\in\mathcal{AMC} \\ |R|< 2^{2nm-m+2}}} \sum_{(X,Y)\in R} [s \text{ is a solution to XOR-Missing-String on } (X,Y)].
\end{aligned}$$

It follows from Lemma 17 that the first summand is at most

$$\sum_{\substack{(R,s)\in\mathcal{AMC} \\ |R|\geq 2^{2nm-m+2}}} (1-2^{-2n-2}) \cdot |R| \leq (1-2^{-4n}) \cdot S.$$

Let $|\mathcal{AMC}|$ denote the size of $\mathcal{AMC}$, then the second summand is at most

$$\begin{aligned}
\sum_{\substack{(R,s)\in\mathcal{AMC} \\ |R|< 2^{2nm-m+2}}} |R| &\leq |\mathcal{AMC}| \cdot 2^{2nm-m+2}. \\
&\leq |\mathcal{AMC}| \cdot S/2^{m-2}. \qquad \text{(since } S \geq 2^{2nm}\text{)}
\end{aligned}$$

If $|\mathcal{AMC}| \leq 2^{m/2}$, then the second summand is at most $2^{-m/2+2} \cdot S$, which is at most $2^{-8n} \cdot S$ since $m \geq 20n$. Summing everything together, we have that

$$C_{\text{correct}} \leq (1-2^{-4n}+2^{-8n}) \cdot S,$$

which contradicts the previous bound of $C_{\text{correct}} \geq (1-2^{-5n}) \cdot S$. Hence it must be the case that $|\mathcal{AMC}| > 2^{m/2}$. ◀

Lower bounds for pseudodeterministic protocols. A consequence of Theorem 4 is that it also applies to *pseudodeterministic* PostBPP communication protocols for XOR-Missing-String with constant error (say 1/3), as the error probability for such protocols can always be

reduced to as small as possible by running it multiple times and outputting the majority answer. In contrast, the error probability of an arbitrary (non-pseudodeterministic) protocol cannot be amplified in general, since it is unclear how to verify whether a string is a valid solution to XOR-Missing-String.

▷ Claim 21. Let $n \geq 1, 1 \leq m < 2^{n/2}$ be integers. Let Π be a pseudodeterministic PostBPP communication protocol that solves XOR-Missing-String(n, m) with error $1/3$ and has communication complexity C. Then for any $k \geq 1$, there exists a pseudodeterministic PostBPP communication protocol Π_k that solves XOR-Missing-String(n, m) with error $2^{-\Theta(k)}$ and has communication complexity $O(C \cdot k)$.

By setting $k = \Theta(n)$ in the above lemma and applying Theorem 4, we have:

▶ **Corollary 22.** *Let $n \geq 1$ and $20n \leq m < 2^{n/2}$ be integers. Any pseudodeterministic* PostBPP *communication protocol that solves* XOR-Missing-String(n, m) *with error probability $\leq 1/3$ requires communication complexity $\Omega(m/n)$.*

3.2 Proving the Barrier

▶ **Theorem 3.** *There exists an oracle A and its multilinear extension $\widetilde{A}$ such that*

$$\text{pr-PostBPE}^{\widetilde{A}} \subseteq \text{i. o.-SIZE}^{A}[O(n)].$$

Proof of Theorem 3. Let $M_1, M_2, \ldots$ be a syntactic enumeration of PostBPE algorithms each running in time 2^n (that is, each M_i may or may not satisfy the PostBPP promise). If we can prove a barrier for all such algorithms, then we can also prove a barrier for all algorithms running in time $2^{O(n)}$ by a padding argument. We assume that each machine M_i appears infinitely many times in the list $(M_i)_{i \in \mathbb{N}}$.

Let n_1 be a large enough constant and set $n_i = 4^{n_{i-1}}$ for every $i > 1$. Let $m_i = n_i^{20}$ for every $i \geq 1$. Recall that logarithms are always base 2. The oracle A that we construct will have the following structure: On input strings whose lengths are not of the form $2 \log m_i$, the value of A will always be zero. For every $i \geq 1$, the truth table of A on input length $2 \log m_i$ will encode an instance (X_i, Y_i) of XOR-Missing-String(n_i, m_i). More specifically, the truth table of $A \cap \{0,1\}^{2 \log m_i}$ restricted to inputs whose first bit is 0 (resp. 1) will be equal to X_i (resp. Y_i) padded with zeros. Note that the length of (X_i, Y_i) is $2m_i \cdot n_i \leq 2^{2 \log m_i}$.

For every $i \geq 1$, the instance (X_i, Y_i) is designed to diagonalize against $M_i^{\widetilde{A}}$. More precisely, we want the following property to hold: let $\mathcal{GOOD}_i$ be the set of inputs $x \in \{0,1\}^{\log n_i}$ such that the execution of $M_i^{\widetilde{A}}(x)$ satisfies the semantics of PostBPP, then there exists a *non-solution* $\alpha \in \{0,1\}^{n_i}$ of the XOR-Missing-String instance (X_i, Y_i) such that $\alpha_x = M_i^{\widetilde{A}}(x)$ for every $x \in \mathcal{GOOD}_i$. Note that $\alpha = (X_i)_a \oplus (Y_i)_b$ for some $a, b \in [m_i]$, hence by hardwiring a and b we can construct an A-oracle circuit of size $O(\log m_i) \leq O(\log n_i)$ whose truth table is equal to α, and it follows that the pr-PostBPE problem defined by $M_i^{\widetilde{A}}$ on input length $\log n_i$ can be computed by linear-size A-oracle circuits. Therefore, it suffices to satisfy the aforementioned property.

Note that, when considering the truth table of $M_i^{\widetilde{A}}$ on input length $\log n_i$, the algorithm $M_i^{\widetilde{A}}$ can only access $\widetilde{A}$ on inputs of length $\leq n_i$. Therefore, we only need to show that $M_i^{\widetilde{A}_{\leq n_i}}$ fails to solve (X_i, Y_i). Since $2 \log m_{i+1} > n_i$, this implies that the algorithm only sees $(X_{\leq i}, Y_{\leq i})$.

We construct our oracle inductively. Suppose that we have constructed (X_j, Y_j) for every $j < i$, and we need to construct (X_i, Y_i). Let P_i be the following PostBPP communication protocol that tries to solve XOR-Missing-String(n_i, m_i). Given an instance (X, Y), define

an oracle A such that $(X_{<i}, Y_{<i})$ are the previously fixed values and $(X_i, Y_i) = (X, Y)$. Then, Alice and Bob output the "truth table" of $M_i^{\widetilde{A}_{\leq n_i}}$ on inputs of length $\log n_i$, denoted as $\mathbf{tt} \in \{0,1\}^{n_i}$: for every $x \in \{0,1\}^{\log n_i}$, Alice and Bob simulate the computation of $M_i^{\widetilde{A}_{\leq n_i}}(x)$ using (the PostBPP version of) Theorem 15 repeatedly for $(1000n_i + 1)$ times, and output the majority outcome as the x-th bit of $\mathbf{tt}$. (We use boldface to emphasize that $\mathbf{tt}$ is a random variable). Let $\mathcal{GOOD}_i(X, Y)$ denote the set of inputs $x \in \{0,1\}^{\log n_i}$ such that the computation of $M_i^{\widetilde{A}_{\leq n_i}}(x)$ satisfies PostBPP promise, and let $f_i : \mathcal{GOOD}_i(X, Y) \to \{0,1\}$ denote the promise problem defined by $M_i^{\widetilde{A}_{\leq n_i}}$:

$$\mathcal{GOOD}_i^b(X,Y) = \left\{ x \in \{0,1\}^{\log n_i} : \Pr\left[M_i^{\widetilde{A}_{\leq n_i}}(x) = b \;\middle|\; M_i^{\widetilde{A}_{\leq n_i}}(x) \neq \perp \right] \geq 2/3 \right\},$$

$$\mathcal{GOOD}_i(X,Y) = \mathcal{GOOD}_i^0(X,Y) \cup \mathcal{GOOD}_i^1(X,Y),$$

$$f_i(x) = \begin{cases} 0 & \text{if } x \in \mathcal{GOOD}_i^0(X,Y); \\ 1 & \text{if } x \in \mathcal{GOOD}_i^1(X,Y). \end{cases}$$

We say that $\mathbf{tt}$ is *consistent* with f_i if for every $x \in \mathcal{GOOD}_i(X, Y)$, $\mathbf{tt}_x = f_i(x)$. Since the computation of $M_i^{\widetilde{A}_{\leq n_i}}(x)$ is repeated $(1000n_i + 1)$ times, the probability that $\mathbf{tt}$ is consistent with f_i is at least $1 - 2^{-10n_i}$. On the other hand, note that the behavior of P_i is the same as some PostBPP algorithm that has oracle access to $\widetilde{A}$ and runs in time $O(n_i^3)$ (since it considers n_i possible inputs, simulates M_i for $O(n_i)$ times on each of them, and each simulation takes $O(n_i)$ time), hence Theorem 15 implies that P_i can be implemented by a PostBPP communication protocol of complexity $O(n_i^9) = o(m_i)$. It follows from Theorem 4 that P_i cannot solve XOR-Missing-String(n_i, m_i) with success probability more than $1 - 2^{-5n_i}$; in other words, there exists some (X, Y) such that $\mathbf{tt}$ is a non-solution of (X, Y) w.p. at least 2^{-5n_i}. By a union bound, there is a non-zero probability that both of the following hold simultaneously: $\mathbf{tt}$ is consistent with f_i and $\mathbf{tt}$ is a non-solution of (X, Y). This means that there is a non-solution $\alpha \in \{0,1\}^{n_i}$ of (X, Y) such that for every $x \in \mathcal{GOOD}_i(X, Y)$, $M_i^{\widetilde{A}_{\leq n_i}}(x) = \alpha_x$. Setting $(X_i, Y_i) = (X, Y)$, this establishes the property we want for i, and we can continue our construction for $i + 1$. ◀

Due to space constraints, we omit the almost-everywhere algebrization barrier for BPE (Theorem 5) and the half-exponential barrier for Rob-MA$_{\mathsf{E}}$ (Theorem 7) and refer the interested reader to the full version of our paper.

References

1. Scott Aaronson. Quantum computing, postselection, and probabilistic polynomial-time. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 461(2063):3473–3482, 2005. doi:10.1098/rspa.2005.1546.
2. Scott Aaronson. Oracles are subtle but not malicious. In *Conference on Computational Complexity* (CCC), pages 340–354. IEEE Computer Society, 2006. doi:10.1109/CCC.2006.32.
3. Scott Aaronson and Avi Wigderson. Algebrization: A new barrier in complexity theory. *ACM Trans. Comput. Theory*, 1(1):2:1–2:54, 2009. doi:10.1145/1490270.1490272.
4. Eric Allender. Oracles versus proof techniques that do not relativize. In *SIGAL International Symposium on Algorithms*, volume 450 of *Lecture Notes in Computer Science*, pages 39–52. Springer, 1990. doi:10.1007/3-540-52921-7_54.
5. Sanjeev Arora and Boaz Barak. *Computational Complexity - A Modern Approach*. Cambridge University Press, 2009. doi:10.1017/CBO9780511804090.

6 Sanjeev Arora, Russell Impagliazzo, and Umesh Vazirani. Relativizing versus nonrelativizing techniques: the role of local checkability. *Manuscript*, 1992. URL: https://people.eecs.berkeley.edu/~vazirani/pubs/relativizing.pdf.

7 Barış Aydınlıoğlu and Eric Bach. Affine relativization: Unifying the algebrization and relativization barriers. *ACM Trans. Comput. Theory*, 10(1):1:1–1:67, 2018. doi:10.1145/3170704.

8 László Babai, Lance Fortnow, and Carsten Lund. Non-deterministic exponential time has two-prover interactive protocols. *Comput. Complex.*, 1:3–40, 1991. doi:10.1007/BF01200056.

9 László Babai, Lance Fortnow, Noam Nisan, and Avi Wigderson. BPP has subexponential time simulations unless EXPTIME has publishable proofs. *Computational Complexity*, 3:307–318, 1993. doi:10.1007/BF01275486.

10 Theodore P. Baker, John Gill, and Robert Solovay. Relativizations of the P =?NP question. *SIAM J. Comput.*, 4(4):431–442, 1975. doi:10.1137/0204037.

11 Nader H. Bshouty, Richard Cleve, Ricard Gavaldà, Sampath Kannan, and Christino Tamon. Oracles and queries that are sufficient for exact learning. *J. Comput. Syst. Sci.*, 52(3):421–433, 1996. doi:10.1006/jcss.1996.0032.

12 Harry Buhrman, Lance Fortnow, and Thomas Thierauf. Nonrelativizing separations. In *CCC*, pages 8–12, 1998. doi:10.1109/CCC.1998.694585.

13 Jin-yi Cai. $\mathsf{S}_2^p \subseteq \mathsf{ZPP}^{\mathsf{NP}}$. *J. Comput. Syst. Sci.*, 73(1):25–35, 2007. doi:10.1016/j.jcss.2003.07.015.

14 Jin-yi Cai, Venkatesan T. Chakaravarthy, Lane A. Hemaspaandra, and Mitsunori Ogihara. Competing provers yield improved Karp–Lipton collapse results. *Inf. Comput.*, 198(1):1–23, 2005. doi:10.1016/j.ic.2005.01.002.

15 Lijie Chen, Shuichi Hirahara, and Hanlin Ren. Symmetric exponential time requires near-maximum circuit size. In *STOC*, pages 1990–1999. ACM, 2024. doi:10.1145/3618260.3649624.

16 Lijie Chen, Jiatu Li, and Jingxun Liang. Maximum circuit lower bounds for exponential-time arthur merlin. In *STOC*, pages 1348–1358. ACM, 2025. doi:10.1145/3717823.3718224.

17 Lijie Chen, Zhenjian Lu, Igor C. Oliveira, Hanlin Ren, and Rahul Santhanam. Polynomial-time pseudodeterministic construction of primes. In *FOCS*, pages 1261–1270. IEEE, 2023. doi:10.1109/FOCS57990.2023.00074.

18 Lijie Chen, Dylan M. McKay, Cody D. Murray, and R. Ryan Williams. Relations and equivalences between circuit lower bounds and Karp–Lipton theorems. In *CCC*, volume 137 of *LIPIcs*, pages 30:1–30:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.CCC.2019.30.

19 Lance Fortnow. The role of relativization in complexity theory. *Bull. EATCS*, 52:229–243, 1994.

20 Eran Gat and Shafi Goldwasser. Probabilistic search algorithms with unique answers and their cryptographic applications. *Electronic Colloquium on Computational Complexity* (ECCC), 18:136, 2011. URL: https://eccc.weizmann.ac.il/report/2011/136.

21 Oded Goldreich. *Computational complexity - a conceptual perspective*. Cambridge University Press, 2008. doi:10.1017/CBO9780511804106.

22 Mika Göös, Toniann Pitassi, and Thomas Watson. The landscape of communication complexity classes. *Comput. Complex.*, 27(2):245–304, June 2018. doi:10.1007/s00037-018-0166-6.

23 Yenjo Han, Lane A. Hemaspaandra, and Thomas Thierauf. Threshold computation and cryptographic security. *SIAM J. Comput.*, 26(1):59–78, 1997. doi:10.1137/S0097539792240467.

24 Shuichi Hirahara, Zhenjian Lu, and Hanlin Ren. Bounded relativization. In *CCC*, volume 264 of *LIPIcs*, pages 6:1–6:45. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.CCC.2023.6.

25 Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova. An axiomatic approach to algebrization. In *STOC*, pages 695–704. ACM, 2009. doi:10.1145/1536414.1536509.

26 Russell Impagliazzo, Valentine Kabanets, and Ilya Volkovich. The power of natural properties as oracles. In *Computational Complexity Conference* (CCC), volume 102 of *LIPIcs*, pages 7:1–7:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2018. doi:10.4230/LIPIcs.CCC.2018.7.

27 Ravi Kannan. Circuit-size lower bounds and non-reducibility to sparse sets. *Inf. Control.*, 55(1-3):40–56, 1982. doi:10.1016/S0019-9958(82)90382-5.

28 Richard M. Karp and Richard J. Lipton. Some connections between nonuniform and uniform complexity classes. In *Symposium on Theory of Computing* (STOC), pages 302–309. ACM, 1980. doi:10.1145/800141.804678.

29 H. Klauck. Rectangle size bounds and threshold covers in communication complexity. In *18th IEEE Annual Conference on Computational Complexity, 2003. Proceedings.*, pages 118–134, 2003. doi:10.1109/CCC.2003.1214415.

30 Robert Kleinberg, Oliver Korten, Daniel Mitropolsky, and Christos Papadimitriou. Total functions in the polynomial hierarchy. In *ITCS*, volume 185 of *LIPIcs*, pages 44:1–44:18, 2021. doi:10.4230/LIPIcs.ITCS.2021.44.

31 Johannes Köbler and Osamu Watanabe. New collapse consequences of NP having small circuits. *SIAM J. Comput.*, 28(1):311–324, 1998. doi:10.1137/S0097539795296206.

32 Oliver Korten. The hardest explicit construction. In *Symposium on Foundations of Computer Science* (FOCS), pages 433–444. IEEE, 2021. doi:10.1109/FOCS52979.2021.00051.

33 Zeyong Li. Symmetric exponential time requires near-maximum circuit size: Simplified, truly uniform. In *STOC*, pages 2000–2007. ACM, 2024. doi:10.1145/3618260.3649615.

34 Zhenjian Lu, Igor C. Oliveira, Hanlin Ren, and Rahul Santhanam. On the complexity of avoiding heavy elements. In *FOCS*, pages 2403–2412. IEEE, 2024. doi:10.1109/FOCS61266.2024.00140.

35 Carsten Lund, Lance Fortnow, Howard J. Karloff, and Noam Nisan. Algebraic methods for interactive proof systems. *J. ACM*, 39(4):859–868, 1992. doi:10.1145/146585.146605.

36 Peter Bro Miltersen, N. V. Vinodchandran, and Osamu Watanabe. Super-polynomial versus half-exponential circuit size in the exponential hierarchy. In *COCOON*, volume 1627 of *Lecture Notes in Computer Science*, pages 210–220. Springer, 1999. doi:10.1007/3-540-48686-0_21.

37 Alexander A. Razborov and Steven Rudich. Natural proofs. *J. Comput. Syst. Sci.*, 55(1):24–35, 1997. doi:10.1006/JCSS.1997.1494.

38 Hanlin Ren, Rahul Santhanam, and Zhikun Wang. On the range avoidance problem for circuits. In *FOCS*, pages 640–650. IEEE, 2022. doi:10.1109/FOCS54457.2022.00067.

39 Rahul Santhanam. Circuit lower bounds for Merlin–Arthur classes. *SIAM J. Comput.*, 39(3):1038–1061, 2009. doi:10.1137/070702680.

40 Adi Shamir. IP = PSPACE. *J. ACM*, 39(4):869–877, 1992. doi:10.1145/146585.146609.

41 Larry J. Stockmeyer and Albert R. Meyer. Cosmological lower bound on the circuit complexity of a small problem in logic. *J. ACM*, 49(6):753–784, 2002. doi:10.1145/602220.602223.

42 N. V. Vinodchandran. A note on the circuit complexity of PP. *Theor. Comput. Sci.*, 347(1-2):415–418, 2005. doi:10.1016/j.tcs.2005.07.032.

43 Nikhil Vyas and R. Ryan Williams. On oracles and algorithmic methods for proving lower bounds. In *Innovations in Theoretical Computer Science Conference* (ITCS), volume 251 of *LIPIcs*, pages 99:1–99:26. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.ITCS.2023.99.

44 Christopher B. Wilson. Relativized circuit complexity. *J. Comput. Syst. Sci.*, 31(2):169–181, 1985. doi:10.1016/0022-0000(85)90040-6.