Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 25 (2019)

On Nonadaptive Security Reductions of Hitting Set Generators

Shuichi Hirahara*
National Institute of Informatics

Osamu Watanabe†
Tokyo Institute of Technology

June 23, 2020

Abstract

One of the central open questions in the theory of average-case complexity is to establish the equivalence between the worst-case and average-case complexity of the Polynomial-time Hierarchy (PH). One general approach is to show that there exists a PH-computable hitting set generator whose security is based on some NP-hard problem. We present the limits of such an approach, by showing that there exists no exponential-time-computable hitting set generator whose security can be proved by using a nonadaptive randomized polynomial-time reduction from any problem outside $\mathsf{AM} \cap \mathsf{coAM}$, which significantly improves the previous upper bound $\mathsf{BPP}^{\mathsf{NP}}$ of Gutfreund and Vadhan (RANDOM/APPROX 2008 [GV08]). In particular, any security proof of a hitting set generator based on some NP-hard problem must use either an adaptive or non-black-box reduction (unless the polynomial-time hierarchy collapses). To the best of our knowledge, this is the first result that shows limits of black-box reductions from an NP-hard problem to some form of a distributional problem in DistPH.

Based on our results, we argue that the recent worst-case to average-case reduction of Hirahara (FOCS 2018 [Hir18]) is inherently non-black-box, without relying on any unproven assumptions. On the other hand, combining the non-black-box reduction with our simulation technique of black-box reductions, we exhibit the existence of a "non-black-box selector" for GapMCSP, i.e., an efficient algorithm that solves GapMCSP given as advice two circuits one of which is guaranteed to compute GapMCSP.

*s_hirahara@nii.ac.jp
†watanabe@c.titech.ac.jp

ISSN 1433-8092

Contents

1 Introduction

The technique of reductions is one of central tools in complexity theory. In order to show that a computational task A is easier than another computational task B, it suffices to design a (black-box) reduction, i.e., the algorithm that solves A given oracle access to B. Most reductions of complexity theory are *black-box*. That is, the correctness of a reduction can be established without assuming any computational efficiency of the oracle. Black-box reductions are quite powerful and led us to, for instance, the discovery of thousands of NP-complete problems computationally equivalent to each other. However, a line of work has exhibited limits of black-box reductions: Black-box reductions are too general to resolve several important open questions. We herein continue the study of black-box reductions especially in the context of the construction of a hitting set generator.

A *hitting set generator* γ-secure against a class $\mathcal{C}$ is a family of functions $G = \{G_\ell : \{0,1\}^{s(\ell)} \to \{0,1\}^\ell\}_{\ell\in\mathbb{N}}$ such that no $\mathcal{C}$-algorithm can γ-avoid G; here we say that an algorithm R *γ-avoids* G if R rejects every string in the image of G, and R accepts at least a γ-fraction of all inputs of length ℓ for every $\ell \in \mathbb{N}$. By default, we assume $\gamma := 1/4$ and we say that R *avoids* G if R $(1/4)$-avoids G. A typical approach for constructing a hitting set generator is to design a black-box reduction that reduces some computationally hard task to the task of avoiding a hitting set generator.

In fact, there have been already several known proof techniques that are not black-box. Impagliazzo and Wigderson [IW01] constructed a hitting set generator based on the uniform hardness assumption that EXP $\neq$ BPP. Their security proof of the hitting set generator is not a (black-box) reduction from the task of solving EXP to the task of avoiding the hitting set generator; they crucially exploited the fact that there exists an efficient algorithm that avoids the hitting set generator. Trevisan and Vadhan [TV07] and Gutfreund and Vadhan [GV08] showed that the security reduction of [IW01] is inherently non-black-box in some senses. More recently, building on [CIKK16, HS17], Hirahara [Hir18] applied the proof techniques for constructing a hitting set generator to the context of average-case complexity, and presented the first non-black-box worst-case to average-case reduction within NP.

Given the fact that there are already non-black-box proof techniques, why should we study the limits of black-box reductions? We highlight several points:

1. Black-box reductions are *more general* and *useful* than non-black-box reductions. Therefore, it is desirable to have a black-box reduction when it is possible; studying limits of black-box reductions enables us to identify when one can hope to construct a black-box reduction.

   For example, Impagliazzo and Wigderson [IW01] showed that EXP $\not\subseteq$ BPP implies that BPP can be derandomized in *sub-exponential* time (on most inputs, for infinitely many input lengths). This is shown by a non-black-box reduction, and it is not known whether the result can be generalized to a "high-end" result: does EXP $\not\subseteq$ BPSUBEXP imply that BPP can be derandomized in *quasi-polynomial* time? On one hand, Trevisan and Vadhan [TV07] used a black-box reduction and provided a positive answer to this question when EXP is replaced with PSPACE. On the other hand, Gutfreund and Vadhan [GV08] showed that a (mildly adaptive) black-box reduction cannot be used to prove the "high-end" result for EXP.

2. Studying limits of black-box reductions can inspire new black-box reductions. Inspired by this work, Hirahara [Hir20c, Hir20b] subsequently presented new constructions of black-box reductions to Kolmogorov complexity, which were previously conjectured to be impossible.

3. Surprisingly, in some cases, the proof techniques for showing limits of black-box reductions can be combined with non-black-box reductions. We will show that one of our new algorithms for simulating black-box reductions can be combined with a non-black-box reduction of [Hir18] (under some assumptions), and present a new structural property of an approximation version of the Minimum Circuit Size Problem (MCSP [KC00]).

As a main result of this paper, we show that any security proof of a hitting set generator based on some NP-hard problem must use either an adaptive or non-black-box reduction. This is the first limit of black-box worst-case to average-case reductions from NP-hard problems to some form of a distributional problem in DistPH.

Due to the connection to several research areas such as average-case complexity, black-box reductions, and derandomization, it is not easy to describe the literature in few words; we thus review the literature in the subsequent two sections. In Section 2, we review the theory of average-case complexity and state our main results. In Section 3, we review the non-black-box reduction of [Hir18], present some applications of our results, and describe our proof techniques.

2 Average-Case Complexity

2.1 Background

One of the central open questions in the theory of average-case complexity [Lev86] is to establish the equivalence between the worst-case and average-case complexity of NP.

Open Question 1. Does $\mathsf{DistNP} \subseteq \mathsf{AvgP}$ imply $\mathsf{NP} = \mathsf{P}$?

Here DistNP is the class of distributional problems $(L, \mathcal{D})$ (i.e., a pair of a problem and its input distribution) such that $L \in \mathsf{NP}$ and $\mathcal{D}$ is an efficiently samplable distribution. AvgP is the class of distributional problems that admit an errorless heuristic polynomial-time scheme [BT06a] (also known as an "average-case polynomial-time algorithm"). Here, for $L \subseteq \{0,1\}^*$ and $\mathcal{D} = \{\mathcal{D}_m\}_{m \in \mathbb{N}}$, a distributional problem $(L, \mathcal{D})$ is said to be in AvgP if there exists an algorithm M such that, for every $m \in \mathbb{N}$, given an input x in the support of D_m, and a parameter $\delta > 0$,

1. $M(x, \delta)$ halts in time $\mathsf{poly}(m, 1/\delta)$,
2. $M(x, \delta)$ outputs either the correct answer $L(x)$ or $\perp$ ("I don't know"), and
3. the probability that $M(x, \delta)$ outputs $\perp$ over a choice of $x \sim \mathcal{D}_m$ is at most δ.

Open Question 1 is of particular importance from the perspective of cryptography: Average-case hardness of NP is a prerequisite for constructing secure complexity-theoretic cryptographic primitives such as one-way functions (OWFs). Thus resolving Open Question 1 is an important step towards building cryptographic primitives whose security is based on more plausible assumptions (e.g., the worst-case hardness of NP).

There has been a line of work showing that Open Question 1 cannot be resolved by using either relativizing proof techniques [Imp11, Wat12], black-box worst-case-to-average-case reductions [FF93, BT06b, AGGM06, BB15, ABX08], or error-correcting codes [Vio05].

For large enough complexity classes such as PSPACE and EXP, there is a general technique for converting any worst-case hard function f to some two-sided-error average-case hard function

$\mathrm{Enc}(f)$ based on error-correcting codes [STV01, TV07]. Here, the encoded function $\mathrm{Enc}(f)$ is computable in EXP or PSPACE given oracle access to f; thus, the worst-case and average-case complexity of such large complexity classes are known to be equivalent. Viola [Vio05] showed limits of such an approach: $\mathrm{Enc}(f)$ cannot be computed in PH^f; thus, the proof technique of using error-correcting codes is not sufficient to resolve Open Question 1 as well as the following weaker open question:

Open Question 2. Does $\mathsf{DistPH} \subseteq \mathsf{AvgP}$ imply $\mathsf{PH} = \mathsf{P}$ (or, equivalently, $\mathsf{NP} = \mathsf{P}$)? [1]

Note that Open Question 2 is an easier question than Open Question 1, since $\mathsf{PH} = \mathsf{P}$ is known to be equivalent to $\mathsf{NP} = \mathsf{P}$. In fact, this well-known equivalence between $\mathsf{PH} = \mathsf{P}$ and $\mathsf{NP} = \mathsf{P}$ is shown by using a non-black-box reduction technique[2]; as we will explain later, this is one reason why all the previous limits of black-box reductions [FF93, BT06b, AGGM06, BB15, ABX08] fail to explain the difficulty of resolving Open Question 2. In this work, we present the first limit of black-box reduction techniques for resolving Open Question 2, thereby clarifying what kind of proof techniques are useful. We emphasize that, while Viola's result [Vio05] excludes the construction of error-correcting codes within PH, it does not show limits of black-box worst-case to average-case reduction techniques such as Ajtai's reduction [Ajt96].

One general approach for constructing an (errorless) average-case hard function is to make use of a hitting set generator. Indeed, a hitting set generator G secure against polynomial-time algorithms naturally induces a hard distributional problem in DistNP^G: Consider the distributional problem $(\mathrm{Im}(G), \mathcal{U})$, i.e., the distributional problem of checking whether an input x is in the image of G, where x is randomly chosen from the uniform distribution $\mathcal{U}$. Since the number of YES instances of $\mathrm{Im}(G)$ is small under the uniform distribution, any errorless heuristic algorithm must reject a large fraction of NO instances, which gives rise to an algorithm that avoids G. To summarize:

Fact 3 (Implicit in [Hir18])**.** *Suppose there exists a hitting set generator* $G := \{G_\ell \colon \{0,1\}^{\ell-1} \to \{0,1\}^\ell\}_{\ell \in \mathbb{N}}$ *that is* $1/4$*-secure against polynomial-time algorithms. Then,* $\mathsf{DistNP}^G \not\subseteq \mathsf{AvgP}$*. In particular, when* G *is computable in* PH*, we obtain* $\mathsf{DistPH} \not\subseteq \mathsf{AvgP}$*.*

Fact 3 suggests an approach for resolving Open Question 2: Try to construct a PH-computable hitting set generator whose security is based on the worst-case hardness of NP. How do we compare this approach with the technique based on error-correcting codes [Vio05]? Our approach is *more general*, because, given a two-sided-error average-case hard function $\mathrm{Enc}(f)$, one can construct a pseudorandom generator $G = \{G_\ell \colon \{0,1\}^{\ell-1} \to \{0,1\}^\ell\}_{\ell \in \mathbb{N}}$ defined as $G_\ell(z) := (z, \mathrm{Enc}(f)(z))$ for a seed $z \in \{0,1\}^{\ell-1}$ [Yao82].

In order to construct a secure hitting set generator based on the hardness of a problem L, we need to argue that, if there exists an efficient algorithm that avoids G, then L can be solved efficiently. A typical way to establish such an implication is to design *black-box reductions* from L to a distinguisher for a hitting set generator. Specifically, for a candidate hitting set generator G, a reduction M is said to be a black-box reduction from L to any γ-avoiding oracle R for G if, for every input x and any oracle R that γ-avoids G, M computes L on input x under the oracle R.

Gutfreund and Vadhan [GV08] initiated the study of limits of such a black-box reduction, motivated by the question on whether derandomization is possible under uniform assumptions

[1] We mention in passing that Pavan, Santhanam, and Vinodchandran [PSV06] made some progress, by proving that $\mathsf{DistP}^{\mathsf{NP}} \subseteq \mathsf{AvgP}$ implies $\mathsf{NP} = \mathsf{P}$, under the implausible assumption that $\mathsf{NP} \subseteq \mathsf{P/poly}$.

[2] Indeed, if the reduction is black-box, we should have $\mathsf{PH} \subseteq \mathsf{P}^{\mathsf{NP}}$, which means that PH collapses.

(e.g., [IW01, TV07]). They showed that any polynomial-time randomized nonadaptive black-box reductions to any oracle avoiding an exponential-time-computable hitting set generator G can be simulated in $\mathsf{BPP}^{\mathsf{NP}}$. Unfortunately, their upper bound is too weak to deduce any limit of the approach on Open Question 2 since $\mathsf{NP} \subseteq \mathsf{BPP}^{\mathsf{NP}}$. Similarly, it is impossible to deduce any limit of the approach on Open Question 1, because the upper bound becomes trivial when G is polynomial-time-computable.

2.2 Our Results: Limits of Security Proof of Hitting Set Generators

We significantly improve the upper bound of [GV08] to $\mathsf{AM} \cap \mathsf{coAM}$. We also show upper bounds of $\mathsf{NP/poly} \cap \mathsf{coNP/poly} \cap \mathsf{S}_2^{\mathsf{NP}}$ even if G is not computable.

To state our results formally, let $\mathsf{BPP}_{\|}^R$ denote the class of languages solvable by a randomized polynomial-time machine with nonadaptive oracle access to R.[3] In the definition of a black-box reduction M to any γ-avoiding oracle R, the reduction M is not allowed to depend on R. However, we will show that the existence of a randomized nonadaptive black-box reduction from L to any γ-avoding oracle R is equivalent to saying that $L \in \mathsf{BPP}_{\|}^R$ for every oracle R that γ-avoids R.[4] In light of this, the result of Gutfreund and Vadhan [GV08] can be stated as $\bigcap_R \mathsf{BPP}_{\|}^R \subseteq \mathsf{BPP}^{\mathsf{NP}}$, where the intersection is taken over all oracles R that γ-avoid an exponential-time computable function G. Our main result improves $\mathsf{BPP}^{\mathsf{NP}}$ to $\mathsf{AM} \cap \mathsf{coAM}$:

Theorem 4 (Main). *Let $G = \{G_\ell : \{0,1\}^{s(\ell)} \to \{0,1\}^\ell\}_{\ell \in \mathbb{N}}$ be any (not necessarily computable) family of functions and $\gamma \colon \mathbb{N} \to [0,1)$ be a parameter such that*

- *there exists a constant $\epsilon > 0$ such that $s(\ell) \leq (1-\epsilon)\ell$ for all large $\ell \in \mathbb{N}$, and*
- *there exists a constant $c > 0$ such that $\gamma(\ell) \leq 1 - \ell^{-c}$ for all large $\ell \in \mathbb{N}$.*

Then,

$$\bigcap_R \mathsf{BPP}_{\|}^R \quad \subseteq \quad \mathsf{NP/poly} \cap \mathsf{coNP/poly} \cap \mathsf{S}_2^{\mathsf{NP}},$$

where the intersection is taken over all oracles R that γ-avoids G. Moreover, if G can be computed in time $2^{O(\ell)}$, then we also have

$$\bigcap_R \mathsf{BPP}_{\|}^R \quad \subseteq \quad \mathsf{AM} \cap \mathsf{coAM}.$$

At the core of Theorem 4 is the following two types of algorithms simulating black-box reductions: One is an $\mathsf{S}_2^{\mathrm{p}}$-type algorithm that simulates any query $q \stackrel{?}{\in} R$ of length at most $\Theta(\log n)$, and the other is an $\mathsf{AM} \cap \mathsf{coAM}$-type algorithm that simulates any query $q \stackrel{?}{\in} R$ of length at least $\Theta(\log n)$. In particular, if G is exponential-time computable, the $\mathsf{S}_2^{\mathrm{p}}$-type algorithm can be replaced with a polynomial-time algorithm and obtain the $\mathsf{AM} \cap \mathsf{coAM}$ upper bound.

Theorem 4 shows that there exists no hitting set generator whose security can be based on the hardness of some NP-hard problem via a nonadaptive reduction (unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$). In

[3] The subscript $\|$ stands for parallel queries.

[4] We state our results in the latter way because this makes our impossibility results stronger. The proof of the equivalence can be found in Lemma 14.

particular, the approach for Open Question 2 by constructing a PH-computable hitting set generator based on an NP-hard problem must use either an adaptive or non-black-box reduction.

It is worthy of note that Theorem 4 is almost tight from several perspectives: First, it is impossible to extend Theorem 4 to the case of *adaptive* reductions (unless PSPACE = AM). Indeed, Trevisan and Vadhan [TV07] constructed an exponential-time-computable pseudorandom generator based on the intractability of some PSPACE-complete problem, and its security reduction is adaptive and black-box in the sense of Theorem 4. Second, our $\mathsf{S}_2^{\mathrm{p}}$-type algorithm for simulating short queries is completely tight when G is a universal Turing machine. Third, it is possible to construct a hitting set generator based on the hardness of SZK (Statistical Zero Knowledge), which is one of the best lower bound on AM∩coAM; thus, the AM∩coAM upper bound of Theorem 4 cannot be significantly improved. The reader is referred to Appendix A for the details.

2.3 Related Work: Limits of Worst-case to Average-case Reductions within NP

To the best of our knowledge, Theorem 4 is *the first result* that shows limits of black-box reductions from an NP-hard problem to (some form of) a distributional problem in DistPH. In order to explain this in more detail, we review the previous work on limits of worst-case to average-case reductions within NP.

A natural approach for establishing the equivalence between the worst-case and average-case complexity of NP is by means of black-box reductions. That is, it is sufficient for resolving Open Question 1 to design a reduction that solves some NP-hard problem L, using oracle access to an errorless heuristic algorithm M that solves some distributional problem in DistNP. A line of work has been devoted to explaining why such a black-box reduction technique is too general to establish a worst-case to average-case connection for an NP-complete problem.

Building on the work of Feigenbaum and Fortnow [FF93], Bogdanov and Trevisan [BT06b] showed that if a worst-case problem L is reducible to some distributional problem in DistNP via a nonadaptive black-box randomized polynomial-time reduction, then L must be in NP/poly ∩ coNP/poly. This in particular shows that the average-case hardness of NP cannot be based on the worst-case hardness of an NP-complete problem using such a reduction technique (unless the polynomial-time hierarchy collapses [Yap83]). Akavia, Goldreich, Goldwasser and Moshkovitz [AGGM06, AGGM10] showed that, in the special case of a nonadaptive reduction to the task of inverting a one-way function, the upper bound of [BT06b] can be improved to AM ∩ coAM, thereby removing the advice "/poly". Bogdanov and Brzuska [BB15] showed that even an adaptive reduction to the task of inverting a size-verifiable one-way function cannot be used for any problem outside AM ∩ coAM. Applebaum, Barak, and Xiao [ABX08] studied black-box reductions to PAC learning, and observed that the technique of [AGGM06] can be applied to (some restricted type of) a black-box reduction to the task of inverting an auxiliary-input one-way function (AIOWF), which is a weaker primitive than a one-way function. We summarize the limits of black-box reductions (depicted by $\to$) as well as known implications (depicted by $\Longrightarrow$) in Figure 1.

Compared to the previous results on the limits of black-box worst-case-to-average-case reductions within NP, a surprising aspect of Theorem 4 is that it generalizes to any function G that may not be computable (and this is a key property for obtaining the limits of the approach on Open Question 2). Indeed, almost all the previous results [FF93, BT06b, AGGM06, ABX08] crucially exploit the fact that a verifier can check the correctness of a certificate for an NP problem; thus a dishonest prover can cheat the verifier only in one direction by not providing a certificate for a YES instance. In our simulation algorithms, a verifier cannot compute G and thus cannot prevent

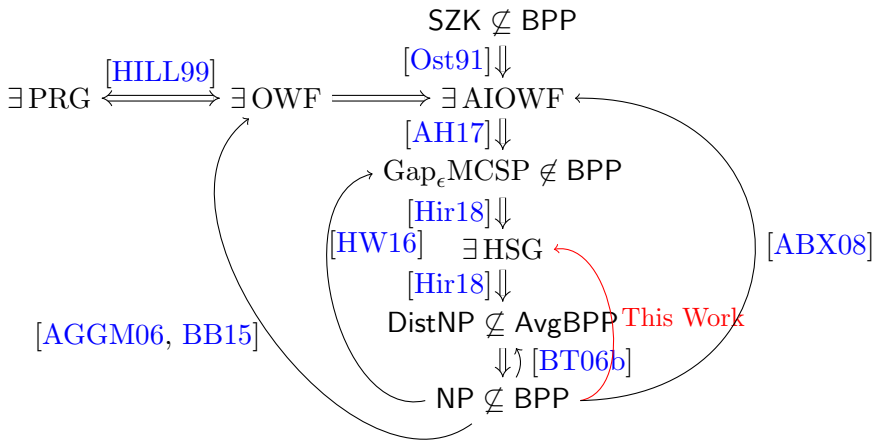

Figure 1: Average-case complexity and limits of black-box reductions. "$A \to B$" means that there is no black-box (or oracle-independent) reduction technique showing "$A \Rightarrow B$" under reasonable complexity theoretic assumptions. The security of all cryptographic primitives is with respect to an almost-everywhere polynomial-time randomized adversary.

dishonest provers from cheating in this way. At a high level, our technical contributions are to overcome this difficulty by combining the ideas of Gutfreund and Vadhan [GV08] with the techniques developed in [FF93, BT06b].

Is it possible to directly deduce some limits of an approach on Open Question 2 from the previous results [FF93, BT06b]? No! Recall that, in order to resolve Open Question 2, it suffices to establish a reduction from an NP-complete problem to DistPH (using the non-black-box equivalence between P = NP and P = PH). The results of [FF93, BT06b] crucially rely on the fact that a YES instance of DistNP is verifiable in polynomial time. If we would like to simulate a black-box reduction to DistNP^A for some oracle A, the simulation protocol of Feigenbaum and Fortnow [FF93] runs in $\mathsf{NP}^A/\mathsf{poly} \cap \mathsf{coNP}^A/\mathsf{poly}$. Thus, in order to simulate a reduction to $\mathsf{Dist}\Sigma_2^{\mathrm{p}} \subseteq \mathsf{DistPH}$, the upper bound becomes $\mathsf{NP}^{\mathsf{NP}}/\mathsf{poly} \cap \mathsf{coNP}^{\mathsf{NP}}/\mathsf{poly}$, which trivially contains NP.

It is also worthy of note that Theorem 4 improves some aspects of all the previous results about limits of black-box reductions within NP. Compared to [BT06b], our results show that the advice "/poly" is not required to simulate black-box reductions to any oracle avoiding an exponential-time-computable hitting set generator. Compared to [AGGM06, ABX08], our results are "improvement" on their results in the sense that the existence of auxiliary-input one-way functions implies the existence of hitting set generators; on the other hand, since the implication goes through the *adaptive* reduction (from the task of inverting a one-way function to a distinguisher for a PRG) of [HILL99], technically speaking, our results are incomparable with their results.[5] Similarly, our

[5] We emphasize that we concern the nonadaptivity of reductions used in the security proof of pseudorandom generators. Several simplified constructions of pseudorandom generators G^f from one-way functions f (e.g., [Hol06, HRV13]) are nonadaptive in the sense that G^f can be efficiently computed with nonadaptive oracle access to f; however, the security reductions of these constructions are adaptive because of the use of Holenstein's uniform hardcore lemma [Hol05]. Similarly, the reduction of [HILL99, Lemma 6.5] is adaptive. (We note that, in the special case when the degeneracy of a one-way function is efficiently computable, the reduction of [HILL99] is nonadaptive.)

results conceptually improve the result of [HW16], but these are technically incomparable, mainly because the implication goes through the non-black-box reduction of [Hir18].

3 In Search of Inherently Non-Black-Box Reduction Techniques

Hirahara [Hir18] presented the first non-black-box worst-case to average-case reduction within NP, which is the motivation for this work. Building on [CIKK16, HS17], Hirahara [Hir18] presented a (nonadaptive) reduction from $\mathrm{Gap}_\epsilon\mathrm{MCSP}$ to a distinguisher for a polynomial-time-computable hitting set generator $G^{\mathsf{int}} = \{G^{\mathsf{int}}_{2^n} \colon \{0,1\}^{\widetilde{O}(2^{\delta n})} \to \{0,1\}^{2^n}\}_{n\in\mathbb{N}}$. Here, G^{int} is a "circuit interpreter": a function that takes a description of a circuit of size $2^{\delta n}$ and outputs its truth table. For a constant $\epsilon > 0$, $\mathrm{Gap}_\epsilon\mathrm{MCSP}$ denotes the problem of approximating the minimum circuit size of a Boolean function $f \colon \{0,1\}^n \to \{0,1\}$ within a factor of $2^{(1-\epsilon)n}$, given the truth table of f. Rudich [Rud97] conjectured that $\mathrm{Gap}_\epsilon\mathrm{MCSP}$ cannot be solved in $\mathsf{coNP/poly}$ (even in the sense of average-case complexity). Therefore, the reduction of [Hir18] is indeed non-black-box under Rudich's conjecture, as otherwise it contradicts the limits of black-box reductions (such as Theorem 4 and [BT06b]).

Here we pose the following question:

> Are the reductions of [Hir18] *inherently* non-black-box? Or should we regard it as an approach for refuting Rudich's conjecture?

On one hand, the proofs of [Hir18] seem to yield only non-black-box reductions, in the sense that the efficiency of an oracle is crucially exploited. On the other hand, if the reduction from $\mathrm{Gap}_\epsilon\mathrm{MCSP}$ to DistNP could be made black-box, by using our coAM simulation protocol of black-box reductions (i.e., Theorem 4), we would obtain $\mathrm{Gap}_\epsilon\mathrm{MCSP} \in \mathsf{coAM} \subseteq \mathsf{coNP/poly}$, which refutes Rudich's conjecture.

In order to answer the question, it is desirable to clarify a fundamental obstacle to applying the simulation techniques of black-box reductions to the reductions of [Hir18], without relying on any unproven assumption.

3.1 Hirahara's Reduction is Unconditionally Non-Black-Box

Based on Theorem 4, we can argue that the reductions of [Hir18] are inherently non-black-box in a certain formal sense. The reason is that the idea of [Hir18] can be applied to not only time-bounded Kolmogorov complexity but also any other types of Kolmogorov complexity, including *resource-unbounded* Kolmogorov complexity. Therefore, if this generalized reduction could be made black-box, then (as outlined below) by Theorem 4 we would obtain a finite-running-time algorithm $\mathsf{S}_2^{\mathsf{NP}}$ that approximates resource-unbounded Kolmogorov complexity, which is a contradiction *unconditionally*.

More specifically, fix any universal Turing machine U, and regard it as a hitting set generator $U = \{U_\ell : \{0,1\}^{\ell/2} \to \{0,1\}^\ell\}_{\ell\in\mathbb{N}}$. That is, U_ℓ takes an input (M, x) of length $\ell/2$, simulates the Turing machine M on input x, and outputs $M(x)$ if the length of the output $M(x)$ is exactly ℓ; otherwise, U_ℓ outputs 1^ℓ.

Claim 5. *Suppose that there exists a computable oracle R that avoids U. Then, there exists a randomized polynomial-time nonadaptive R-oracle algorithm that approximates $\mathrm{K}_U(x)$.*

Proof Sketch. The idea of the non-black-box reduction of [Hir18] is as follows: Given an input $x \in \{0,1\}^n$, take any construction of a pseudorandom generator $G^x \colon \{0,1\}^{\ell/4} \to \{0,1\}^\ell$ based on a worst-case hard function $x \colon \{0,1\}^{\log n} \to \{0,1\}$.[6] For example, we can use the Nisan-Wigderson generator [NW94] combined with some error-correcting codes. The reduction estimates $p := \mathbb{E}_z[R(G^x(z))]$ by sampling, and accepts if and only if p is small.

The correctness is proved as follows: If $\mathrm{K}_U(x) \leq \ell/5$, then $\mathrm{K}_U(G^x(z)) \leq |z| + \mathrm{K}_U(x) \ll \ell/2$ for a large enough ℓ; thus $p = 0$. Conversely, if $p \approx 0$, then by using the security proof of G^x, we obtain a small R-oracle Turing machine that outputs x; thus $\mathrm{K}^R_U(x) \leq \mathsf{poly}(\ell, \log n)$; in particular, *by using the assumption that R is computable*, we obtain $\mathrm{K}_U(x) \leq \mathsf{poly}(\ell, \log n)$. Therefore, the reduction distinguishes the YES instances x such that $\mathrm{K}_U(x) \leq \ell/5$ and the NO instances x such that $\mathrm{K}_U(x) > \mathsf{poly}(\ell, \log n)$. □

Observe that, in the proof above, we crucially used the assumption that R is computable. Can we avoid the assumption and generalize Claim 5 for any R that avoids U? In other words, is there a black-box reduction from approximating $\mathrm{K}_U(x)$ to the task of avoiding U? If it is the case, Theorem 4 implies that approximating $\mathrm{K}_U(x)$ can be done in $\mathsf{S}_2^{\mathsf{NP}}$, which contradicts the undecidability of Kolmogorov complexity. Therefore, we conclude that the reduction of Claim 5 is inherently non-black-box.

3.2 Applications: Non-Black-Box Selector for GapMCSP

As explained in the previous subsection, the non-black-box reductions of [Hir18] cannot be combined with Theorem 4 *unconditionally*. However, we show that our simulation protocol of black-box reductions can be combined with the non-black-box reductions *conditionally*, which constitutes a new structural property of GapMCSP — the existence of a "non-black-box selector."

Theorem 6 (GapMCSP Has a "Non-Black-Box Selector")**.** *For any constant $\epsilon > 0$, there exist some constant $\delta > 0$ and a randomized polynomial-time algorithm that takes as advice two circuits one of which is guaranteed to solve* $\mathrm{Gap}_\epsilon\mathrm{MCSP}$ *and solves* $\mathrm{Gap}_\delta\mathrm{MCSP}$ *with high probability.*

A *selector* for a problem L is an efficient algorithm that solves L given oracle access to two oracles one of which is guaranteed to solve; thus, it "selects" the correct answer from the two oracles. The notion of selector exactly characterizes the class of languages for which advice of logarithmic length can be removed [Hir15]. The selector of Theorem 6 is non-black-box in the sense that it requires to take as advice two polynomial-size circuits instead of black-box access to two oracles.

The main building block of the non-black-box selector is our $\mathsf{S}_2^{\mathrm{p}}$-type simulation algorithm of Theorem 4. Recall that $\mathsf{S}_2^{\mathrm{p}}$ is a proof system where two competing provers, one of which is guaranteed to be honest, try to convince a polynomial-time verifier. In our $\mathsf{S}_2^{\mathrm{p}}$ simulation algorithm of black-box reductions, for each $i \in \{0,1\}$, the ith prover sends a set R_i; the honest prover sends a set R_i that avoids a hitting set generator G. Then a verifier obtains an oracle $R_0 \cap R_1$ that avoids G, to which the reduction is guaranteed to work.

Theorem 6 is proved by combining this $\mathsf{S}_2^{\mathrm{p}}$-type simulation algorithm with the non-black-box reductions of [CIKK16, Hir18].[7] The reason why we can combine the non-black-box reductions with

[6] Here we identify a function with its truth table.

[7] There is an alternative proof based on the search-to-decision reduction of GapMCSP given by Carmosino, Impagliazzo, Kabanets, and Kolokolova [CIKK16]. However, we choose to present the proof by combining the $\mathsf{S}_2^{\mathrm{p}}$-type

our $\mathsf{S}_2^{\mathrm{p}}$-type simulation algorithm is that the non-black-box reduction of [Hir18] is, in fact, a *size-restricted black-box reduction* [GV08]. This is a black-box reduction which works correctly when an oracle can be computed by a polynomial-size circuit. Our $\mathsf{S}_2^{\mathrm{p}}$-type simulation algorithm can simulate the size-restricted black-box reduction under the assumption that there exists a polynomial-size circuit that avoids a hitting set generator.

In contrast, we were not able to combine our $\mathsf{AM} \cap \mathsf{coAM}$ algorithm of Theorem 4 with the non-black-box reductions under similar conditions. We leave it as an interesting open question, which could have an application to fixed-polynomial circuit lower bounds (e.g., [San09]).

Open Question 7 ("Non-Black-Box Instance Checkability" of GapMCSP)**.** Prove that $\mathrm{MCSP} \in \mathsf{P/poly}$ (or $\mathsf{NP} \subseteq \mathsf{P/poly}$) implies $\mathrm{Gap}_\epsilon\mathrm{MCSP} \in \mathsf{coAM}$ for some constant $\epsilon > 0$.

3.3 Our Techniques

We outline our proof strategy for Theorem 4 below. Suppose that we have some reduction M from L to any oracle R that avoids a hitting set generator G. Fix any input $x \in \{0,1\}^*$, and let $\mathcal{Q}_x$ denote the query distribution that a reduction makes on input x. We focus on the case when the length of each query is larger than $\Theta(\log n)$, and explain the proof ideas for showing $L \in \mathsf{AM} \cap \mathsf{coAM}$.

As a warm-up, consider the case when the support $\mathsf{supp}(\mathcal{Q}_x)$ of $\mathcal{Q}_x$ is small (i.e., $|\mathsf{supp}(\mathcal{Q}_x) \cap \{0,1\}^\ell| \ll 2^\ell$ for all large $\ell \in \mathbb{N}$). In this case, we can define an oracle R_1 so that $R_1 := \{0,1\}^* \setminus \mathsf{supp}(\mathcal{Q}_x) \setminus \mathrm{Im}(G)$; this avoids the hitting generator G because $R_1 \cap \mathrm{Im}(G) = \varnothing$ and the size of $R_1 \cap \{0,1\}^\ell$ is at least $2^\ell - |\mathsf{supp}(\mathcal{Q}_x)| - |\mathrm{Im}(G_\ell)| \gg 2^{\ell-1}$ for all large $\ell \in \mathbb{N}$. Therefore, it is guaranteed that the reduction M computes L correctly under the oracle R_1; we can simulate the reduction by simply answering all the queries by saying "No" (since $q \notin R_1$ for every $q \in \mathcal{Q}_x$); hence $L \in \mathsf{BPP}$.

In general, we cannot hope that $\mathsf{supp}(\mathcal{Q}_x)$ is small enough. To generalize the observation above, let us recall the notion of α-heaviness [BT06b]: We say that a query q is *α-heavy* (with respect to $\mathcal{Q}_x$) if the query q is α times more likely to be sampled under $\mathcal{Q}_x$ than the uniform distribution on $\{0,1\}^{|q|}$; that is, $\Pr_{w \sim \mathcal{Q}_x}[w = q] \geq \alpha 2^{-|q|}$. Now we define our new oracle $R_2 := \{0,1\}^* \setminus \{\, q \in \{0,1\}^* \mid q\colon \alpha\text{-heavy} \,\} \setminus \mathrm{Im}(G)$, which can again be shown to avoid G because the fraction of α-heavy queries is at most $1/\alpha$ $(\ll 1)$.

The problem now is that it is difficult to simulate the new oracle R_2; it appears that, given a query q, we need to test whether $q \stackrel{?}{\in} \mathrm{Im}(G)$, which is not possible in $\mathsf{AM} \cap \mathsf{coAM}$. However, it turns out that it is not necessary to test it, as we explain next: Observe that the size of $\mathrm{Im}(G)$ is very small; it is at most $2^{s(\ell)}$ $\left(\ll 2^\ell\right)$. Thus, the probability that a query q is in $\mathrm{Im}(G)$ and q is not α-heavy (i.e., q is rarely queried) is at most $\alpha \cdot 2^{s(\ell)-\ell}$, where ℓ denotes the length of q. As a consequence, the reduction cannot "distinguish" the oracle R_2 and a new oracle $R_3 := \{0,1\}^* \setminus \{\, q \in \{0,1\}^* \mid q\colon \alpha\text{-heavy} \,\}$; hence, we can simulate the reduction if, given a query q, we can decide whether $q \stackrel{?}{\in} R_3$ in $\mathsf{AM} \cap \mathsf{coAM}$.

This task, however, still appears to be difficult for $\mathsf{AM} \cap \mathsf{coAM}$; indeed, at this point, Gutfreund and Vadhan [GV08] used the fact that the approximate counting is possible in $\mathsf{BPP}^{\mathsf{NP}}$, and thereby simulated the oracle R_3 by a $\mathsf{BPP}^{\mathsf{NP}}$ algorithm.

simulation algorithm with the non-black-box reductions in order to highlight the difference between Theorem 6 and Open Question 7.

Our main technical contribution is to develop a way of simulating the reduction to R_3. First, note that the lower bound protocol of Goldwasser and Sipser [GS86] enables us to give an AM certificate for α-heaviness; we can check, given a query q, whether q is $\alpha(1+\epsilon)$-heavy or α-light for any small error parameter $\epsilon > 0$. Thus, we have an AM protocol for $\{0,1\}^* \setminus R_3$ for every query q (except for the queries that are α-heavy and $\alpha(1+\epsilon)$-light).

If, in addition, we had an AM protocol for R_3, then we would be done; however, it does not seem possible in general. The upper bound protocol of Fortnow [For89] performs a similar task, but the protocol can be applied only for a limited purpose: we need to keep the randomness used to generate a query $q \sim \mathcal{Q}_x$ from being revealed to the prover. When the number of queries of the reduction is limited to 1, we can use the upper bound protocol in order to give an AM certificate for R_3; on the other hand, if the reduction makes two queries $(q_1, q_2) \sim \mathcal{Q}_x$, we cannot simultaneously provide AM certificates of the upper bound protocol for *both* q_1 and q_2, because the fact that q_1 and q_2 are sampled *together* may reveal some information about the private randomness. To summarize, the upper bound protocol works only for the *marginal* distribution of each query, but does not work for the *joint* distribution of several queries.

Still, the upper bound protocol is useful for extracting some information about *each* query. For example, the heavy-sample protocol of Bogdanov and Trevisan [BT06b] (which combines the lower and upper bound protocol and sampling) estimates, in $\mathsf{AM} \cap \mathsf{coAM}$, the probability that a query q sampled from $\mathcal{Q}_x$ is α-heavy. This protocol enables us to estimate the probability that $q \in R_3$ over the choice of $q \sim \mathcal{Q}_x$.

The probability that $q \in R_3$ is useful for simulating the reduction M. Feigenbaum and Fortnow [FF93] developed an $\mathsf{AM} \cap \mathsf{coAM}$ protocol that simulates a nonadaptive reduction to an NP oracle R, given as advice the probability that a query q is in R. We generalize this protocol for the case when the oracle R is solvable by AM on average:

Theorem 8 (Generalized Feigenbaum–Fortnow Protocol; informal)**.** *Suppose that M is a randomized polynomial-time nonadaptive reduction to an oracle R whose queries are distributed according to $\mathcal{Q}_x$ on input $x \in \{0,1\}^n$, and that R is solvable by AM on average (i.e., there exists an AM protocol Π_R such that, with probability $1 - 1/\mathsf{poly}(n)$ over the choice of $q \sim \mathcal{Q}_x$, the protocol Π_R computes R on input q). Then, there exists an $\mathsf{AM} \cap \mathsf{coAM}$ protocol Π_M such that, given a probability $p^* \approx \Pr_{q \sim \mathcal{Q}_x}[q \in R]$ as advice, the protocol Π_M simulates the reduction M with probability at least $1 - 1/\mathsf{poly}(n)$.*

Let R denote the complement of R_3, i.e., $R := \{\, q \in \{0,1\}^* \mid q\colon \alpha\text{-heavy} \,\}$. Using the generalized Feigenbaum–Fortnow protocol, we simulate the reduction M to R as follows. Firstly, we use the heavy-sample protocol of [BT06b] in order to estimate $p^* \approx \Pr_{q \sim \mathcal{Q}_x}[q\colon \alpha\text{-heavy}]$. Secondly, using the lower bound protocol of [GS86], we argue that R can be solved by some AM-protocol Π_R on average. Lastly, we use the protocol of Theorem 8 to simulate M. The details can be found in Section 6.

We mention in passing the difficulty of Open Question 7, i.e., the reason why we were not able to combine our $\mathsf{AM} \cap \mathsf{coAM}$-type simulation algorithm with the non-black-box reduction *even conditionally*: The non-black-box reduction outlined in Subsection 3.1 reduces the promise problem whose YES instance consists of $\mathrm{K}_U(x) \leq \ell/5$ and NO instance consists of $\mathrm{K}^R_U(x) > \mathsf{poly}(\ell, \log n)$ to an oracle R. In order to make sure that the promise problem is non-trivial, it is important that R does not depend on x. On the other hand, in our simulation algorithm, we need to choose an oracle R_x depending on the input x, which potentially makes the promise problem trivial. (For example, $\mathrm{K}^{R_x}_U(x)$ may be always close to 0.)

Subsequent Work. Inspired by this work, Allender's conjectures [All12] were refuted under the plausible assumptions about the exponential-time hierarchy [Hir20c, Hir20b]. Moreover, it turned out that the stretch of a hitting set generator construction is important. In [Hir20b], it was shown that there exists a function $G = \{G : \{0,1\}^{n-O(\log n)} \to \{0,1\}^n\}_{n\in\mathbb{N}}$ such that $\mathsf{NEXP} \cup \mathsf{coNEXP} \subseteq \mathsf{BPP}_{\|}^{R}$ for any oracle R that avoids G. This result bypasses our limits of black-box reductions (Theorem 4) because G extends its seed by a small amount of $O(\log n)$ whereas Theorem 4 requires that G extends its seed by a constant factor. In [Hir20a], the approximation quality of non-black-box reductions of [Hir18] is improved. Moreover, based on the improvement, it is shown that, under the assumption that $\mathsf{DistPH} \subseteq \mathsf{AvgP}$, the time-bounded SAT-oracle Kolmogorov complexity of a string x is equal to the time-bounded Kolmogorov complexity of x up to an additive term of $O(\log n)$, for any string $x \in \{0,1\}^n$.

Organization. The rest of this paper is organized as follows. After reviewing necessary background in Section 4, we show that a reduction with only short queries can be simulated by using an $\mathsf{S}_2^{\mathrm{p}}$ algorithm in Section 5. We show how to simulate a reduction with long queries by using an $\mathsf{AM} \cap \mathsf{coAM}$ algorithm in Section 6, which completes our main results. In Section 7, we present a non-black-box selector for GapMCSP.

4 Preliminaries

We identify a language $L \subseteq \{0,1\}^*$ with its characteristic function $L\colon \{0,1\}^* \to \{0,1\}$.. For a language A and $\ell \in \mathbb{N}$, let $A^{=\ell} := A \cap \{0,1\}^\ell$. Let $[n] := \{1,\ldots,n\}$ for $n \in \mathbb{N}$.

We will make use of a standard concentration inequality:

Lemma 9 (Hoeffding's inequality [Hoe63])**.** *For any independent random variables $X_1,\ldots,X_n \in [0,1]$ and any $t \geq 0$, we have $\Pr\left[\left|\sum_{i=1}^n (X_i - \mathbb{E}[X_i])\right| \geq nt\right] \leq 2\exp(-2nt^2)$.*

4.1 Interactive Proof System

AM is the class of languages L that can be accepted by some polynomial-time two-round Arthur-Merlin protocol; that is, there exists a polynomial-time Turing machine V (called an AM verifier) such that

- (Completeness) if $x \in L$ then $\Pr_r[V(x,y,r) = 1 \text{ for some } y] \geq \frac{2}{3}$, and
- (Soundness) if $x \notin L$ then $\Pr_r[V(x,y,r) = 1 \text{ for some } y] \leq \frac{1}{3}$.

For our purpose, it is convenient to use the following characterization of $\mathsf{AM} \cap \mathsf{coAM}$.

Fact 10. *Let $L \subseteq \{0,1\}^*$. Then $L \in \mathsf{AM}\cap\mathsf{coAM}$ if and only if there exists a randomized polynomial-time verifier V of a private-coin constant-round interactive proof system such that, for any input $x \in \{0,1\}^*$,*

- *(Completeness) there exists a prover P such that V outputs $L(x)$ by communicating with P with probability at least $\frac{2}{3}$, and*
- *(Soundness) for any prover P, V outputs $L(x)$ or $\perp$ with P with probability at least $\frac{2}{3}$.*

That is, the verifier outputs a correct answer $L(x)$ when interacting with an honest prover, and it does not output the wrong answer $1-L(x)$ for any cheating prover, with high probability. The fact above follows from the transformation from public coin protocols to private coin protocols [GS86], and the AM hierarchy collapses [Bab85].

Circuits. We will use a circuit in order to make it easy to compose several protocols[8]. Usually, a circuit takes a string of some fixed length as input and outputs a string of some fixed length. For our purpose, it is convenient to extend this usual notion of circuit: By using some encoding, we regard circuits as taking a string of length *at most* l for some $l \in \mathbb{N}$, and outputting a string (not necessarily of fixed length). We regard a circuit as computing a function from $\{0,1\}^*$ to $\{0,1\}^* \cup \{\text{“undefined”}\}$ such that the function outputs “undefined” on inputs of length $> l$.

4.2 Competing Prover System

$\mathsf{S}_2^{\mathrm{p}}$ denotes the class of languages that admit a competing-two-prover system.

Definition 11 ([Can96, RS98])**.** *$\mathsf{S}_2^{\mathrm{p}}$ is the class of languages L such that there exist a polynomial $p(n) = n^{O(1)}$ and a polynomial-time algorithm V such that, for every input $x \in \{0,1\}^*$,*

1. $\exists y \in \{0,1\}^{p(|x|)}, \forall z \in \{0,1\}^{p(|x|)},\ V(x,y,z) = L(x)$, *and*
2. $\exists z \in \{0,1\}^{p(|x|)}, \forall y \in \{0,1\}^{p(|x|)},\ V(x,y,z) = L(x)$.

For an oracle A, the relativized version of $\mathsf{S}_2^{\mathrm{p}}$ is denoted by S_2^A; that is, S_2^A is the class of languages that admit a competing-two-prover system with a polynomial-time A-oracle verifier; $\mathsf{S}_2^{\mathsf{NP}}$ is defined as $\bigcup_{A \in \mathsf{NP}} S_2^A$.

4.3 On Nonadaptive Reductions and Query Distributions

A polynomial-time *nonadaptive* reduction is a polynomial-time oracle Turing machine whose possible queries can be computed without access to an oracle in polynomial time. For simplicity, we assume without loss of generality that, for all inputs of length n, the reduction makes the same number $m = m(n)$ of queries (by adding dummy queries if necessary). For any oracle $R \subseteq \{0,1\}^*$, we denote by $\mathsf{BPP}_{\|}^R$ the class of languages from which there exists a randomized polynomial-time nonadaptive reduction to R. For a nonadaptive reduction M, we denote by $M^R(x)$ the output of the reduction given oracle access to $R \subseteq \{0,1\}^*$ and input $x \in \{0,1\}^*$.

We can modify a randomized nonadaptive reduction M so that the marginal distribution of each query of M is identical; that is, for any query $q \in \{0,1\}^*$, the probability that q is sampled as the ith query of M is the same for all $i \in [m]$. To achieve this, we simply modify M as follows: it generates a permutation $\pi\colon [m] \to [m]$ uniformly at random, runs $M(x)$ to make m queries $q_1, \ldots, q_m$, asks $q_{\pi(i)}$ as the ith query to get an answer a_i from an oracle, and resumes the computation of $M(x)$ to get the decision on x by supplying $a_{\pi^{-1}(1)}, \ldots, a_{\pi^{-1}(m)}$ as oracle answers. It is then easy to see that in the new query machine the ith query distribution is identical for all $i \in [m]$. By the modification above, we can take a single query distribution $\mathcal{Q}_x$ such that each query of $M(x)$ is distributed according to $\mathcal{Q}_x$.

[8] The reader may simply regard a circuit as a Turing machine with an appropriate description to which one can embed some additional information.

Let $\mathcal{Q}$ be a distribution over $\{0,1\}^*$. We use the notation $q \sim \mathcal{Q}$ to indicate that a random variable q is sampled from $\mathcal{Q}$. For a finite set A, we use the same notation $a \sim A$ to indicate that a random variable a is sampled from the uniform distribution over A. For a string $q \in \{0,1\}^*$, let $\mathcal{Q}(q)$ denote $\Pr_{w\sim\mathcal{Q}}[w = q]$. For any parameter $\alpha > 0$, a string $q \in \{0,1\}^*$ of length $\ell \in \mathbb{N}$ is called *α-heavy* (with respect to $\mathcal{Q}$) if $\mathcal{Q}(q) \geq \alpha 2^{-\ell}$; otherwise (i.e., $\mathcal{Q}(q) < \alpha 2^{-\ell}$), it is called *$\alpha$-light*.

4.4 Black-Box Reduction to an Arbitrary Avoiding Oracle

We consider any family of functions $G = \{G_\ell : \{0,1\}^{s(\ell)} \to \{0,1\}^\ell\}_{\ell\in\mathbb{N}}$ that is regarded as a hitting set generator; here, $s\colon \mathbb{N} \to \mathbb{N}$ denotes a function that determines the seed length of the generator. Throughout this paper, we assume that the function $\ell - s(\ell)$ is nonnegative and nondecreasing for simplicity. We measure the time complexity of G with respect to its output length; that is, we say that G is $O(t_G(\ell))$-time computable if there exists a deterministic algorithm that computes, for any given seed $z \in \{0,1\}^{s(\ell)}$ and $\ell \in \mathbb{N}$, the string $G_\ell(z)$ of length ℓ in time $O(t_G(\ell))$. Let $\mathrm{Im}(G_\ell)$ denote the image of G_ℓ; that is, $\mathrm{Im}(G_\ell) = \{\, w \in \{0,1\}^\ell \mid \ w = G_\ell(u) \text{ for some } u \in \{0,1\}^{s(\ell)} \,\}$. Also let $\mathrm{Im}(G) := \cup_{\ell\in\mathbb{N}}\mathrm{Im}(G_\ell)$. We regard G as a hitting set generator, and define the standard notion of avoiding G as follows.

Definition 12. *For any family $G = \{G_\ell : \{0,1\}^{s(\ell)} \to \{0,1\}^\ell\}_{\ell\in\mathbb{N}}$ of functions and any parameter $\gamma\colon \mathbb{N} \to [0,1)$, a set R of strings is said to* γ-avoids G *(or, a γ-avoiding set for G) if*

1. *$R \cap \mathrm{Im}(G) = \varnothing$, and*
2. *$\Pr_{w\sim\{0,1\}^\ell}[\, w \in R \,] \geq \gamma(\ell)$ for all $\ell \in \mathbb{N}$.*

Similarly, for any fixed length $\ell \in \mathbb{N}$, we say that R γ-avoids G at length ℓ *if $R^{=\ell} \cap \mathrm{Im}(G) = \varnothing$ and $\Pr_{w\sim\{0,1\}^\ell}[w \in R] \geq \gamma(\ell)$.*

We always assume that a parameter γ satisfies $\gamma(\ell) \leq 1 - 2^{s(\ell)-\ell}$ for all $\ell \in \mathbb{N}$, as otherwise γ-avoiding sets may not exist. In the context where G is fixed, we omit specifying G and simply say that R is a γ-avoiding set.

Definition 13 (Black-box reduction to γ-avoiding oracles [GV08])**.** *For any family of functions $G = \{G_\ell : \{0,1\}^{s(\ell)} \to \{0,1\}^\ell\}_{\ell\in\mathbb{N}}$ and any language $L \subseteq \{0,1\}^*$, a randomized nonadaptive oracle machine M is called a* black-box reduction from L to any γ-avoiding oracle of G *if, for any γ-avoiding oracle R for G and any $x \in \{0,1\}^*$, we have*

$$\Pr\left[\, M^R(x) = L(x) \,\right] \geq \frac{2}{3}, \tag{1}$$

where the probability is taken over the internal randomness of M.

We emphasize that the definition above requires that there exists a *single* machine that works for *every* γ-avoiding oracle; on the other hand, Theorem 4 states that, if, for every γ-avoiding oracle R, there exists a nonadaptive reduction M_R from L to R, then $L \in \mathsf{AM} \cap \mathsf{coAM}$. That is, the order of the quantifier is reversed; nonetheless, a diagonalization argument enables us to establish the equivalence:

Lemma 14. *Let $G = \{G_\ell : \{0,1\}^{s(\ell)} \to \{0,1\}^\ell\}_{\ell\in\mathbb{N}}$ be a family of functions, $\gamma\colon \mathbb{N} \to [0,1)$ be any parameter, and $L \subseteq \{0,1\}^*$ be a language. The following are equivalent:*

1. $L \in \bigcap_{R\colon \gamma\text{-avoids } G} \mathsf{BPP}_{\|}^{R}$.

2. *There exists a randomized polynomial-time nonadaptive black-box reduction from L to any γ-avoiding oracle of G.*

Proof. The direction from the second item to the first item is obvious. We prove below the contrapositive of the other direction.

Suppose that, for any randomized nonadaptive oracle machine M, there exists some γ-avoiding oracle R_M of G such that $\Pr_M\left[M^R(x) = L(x)\right] < \frac{2}{3}$ for some $x \in \{0,1\}^*$. We claim that there exists some *single* γ-avoiding oracle R of G such that $L \notin \mathsf{BPP}_{\|}^{R}$.

To this end, let $\{M_e\}_{e\in\mathbb{N}}$ be the set of all randomized nonadaptive oracle machines. We will construct some γ-avoiding oracle R_e and input x_e (and $\ell_e \in \mathbb{N}$) by induction on $e \in \mathbb{N}$, so that M_e given oracle R_{e+1} fails to compute L on input x_e; then we will define $R := \bigcup_{e\in\mathbb{N}} R_e$. Let us start with $R_0 := \varnothing$ and $\ell_0 := 0$.

At stage $e \in \mathbb{N}$, we claim that there exists some γ-avoiding oracle $R'_{e+1} \subseteq \{0,1\}^*$ and some input $x_e \in \{0,1\}^*$ such that

- $\Pr\left[M_e^{R'_{e+1}}(x_e) = L(x_e)\right] < \frac{2}{3}$, and
- $q \in R_e$ if and only if $q \in R'_{e+1}$ for any string q of length $< \ell_e$.

Indeed, for any oracle Q, let $Q' := \{\, q \in Q \mid |q| \geq \ell_e \,\} \cup \{\, q \in R_e \mid |q| < \ell_e \,\}$. Consider a randomized nonadaptive oracle machine M'_e such that $M_e'^{Q}$ simulates $M_e^{Q'}$; that is, M'_e is hardwired with the set $\{\, q \in R_e \mid |q| < \ell_e \,\}$, and simulates M_e and answer any query q of length $< \ell_e$ by using the hardwired information. By our assumption, there exists some γ-avoiding oracle $\hat{R}_{e+1}$ of G such that $\Pr\left[M_e'^{\hat{R}_{e+1}}(x_e) = L(x_e)\right] < \frac{2}{3}$ for some $x_e \in \{0,1\}^*$; by the definition of M'_e, we obtain $\Pr\left[M_e^{R'_{e+1}}(x_e) = L(x_e)\right] < \frac{2}{3}$ for $R'_{e+1} := \{\, q \in \hat{R}_{e+1} \mid |q| \geq \ell_e \,\} \cup \{\, q \in R_e \mid |q| < \ell_e \,\}$, which is again γ-avoiding G. This completes the proof of the claim above. Now define $\ell_{e+1} \in \mathbb{N}$ as a large enough integer so that $\ell_{e+1} \geq \ell_e$ and the machine M_e on input x_e does not query any string of length $\geq \ell_{e+1}$, and define an oracle $R_{e+1} := \{\, q \in R'_{e+1} \mid |q| < \ell_{e+1} \,\}$, which completes the construction of stage $e \in \mathbb{N}$.

Define $R := \bigcup_{e\in\mathbb{N}} R_e$, which γ-avoids G by the construction above. By the choice of $(\ell_e)_{e\in\mathbb{N}}$, we have

$$\Pr\left[M_e^{R}(x_e) = L(x_e)\right] = \Pr\left[M_e^{R_{e+1}}(x_e) = L(x_e)\right] < \frac{2}{3},$$

for every randomized nonadaptive oracle machine M_e. Thus $L \notin \mathsf{BPP}_{\|}^{R}$. □

5 Simulating Short Queries by Competing Prover Systems

In this section, we show how to simulate a reduction that makes only short queries in $\mathsf{S}_2^{\mathrm{p}}$.

Theorem 15 ($\mathsf{S}_2^{\mathrm{p}}$ Simulation of Short Queries)**.** *Let $G = \{G_\ell : \{0,1\}^{s(\ell)} \to \{0,1\}^{\ell}\}_{\ell\in\mathbb{N}}$ be any family of functions and $\gamma\colon \mathbb{N} \to [0,1)$ be a parameter such that $\gamma(\ell) \leq 1 - 2^{s(\ell)-\ell+1}$ for all large $\ell \in \mathbb{N}$. Suppose that there exists a randomized polynomial-time black-box reduction M from a language L to any γ-avoiding oracle for G such that the length of any query of M is at most $O(\log n)$ for every input of length n. Then $L \in \mathsf{S}_2^{\mathrm{p}}$.*

Proof. The idea is that two competing provers send the image $\mathrm{Im}(G)$ of G as a certificate. Given two possible images $I_0, I_1 \subseteq \{0,1\}^*$, $R := \{0,1\}^* \setminus I_0 \setminus I_1$ is an avoiding set for G. Moreover, since $|\mathrm{Im}(G)|$ is small, the set R is dense enough. We then derandomize a BPP computation by using the power of $\mathsf{S}_2^{\mathrm{p}}$.[9] (Recall that $\mathsf{BPP} \subseteq \mathsf{S}_2^{\mathrm{p}}$ [Can96, RS98].) Details follow.

Let $c \log n$ be an upper bound on the length of queries that M makes. Our $\mathsf{S}_2^{\mathrm{p}}$ algorithm is as follows: Fix any input x of length n. We number the two competing provers 0 and 1. The ith prover ($i \in \{0,1\}$) sends, for each $\ell \leq c \log n$, a subset $I_{i,\ell} \subseteq \{0,1\}^\ell$ of size at most $2^{s(\ell)}$; an honest prover sets $I_{i,\ell} := \mathrm{Im}(G_\ell)$. Define $I_i := \bigcup_{\ell \leq c \log n} I_{i,\ell}$. Note that such subsets can be encoded as a string of polynomial length. Each prover also sends a list of randomness $r_1^i, \cdots, r_t^i \in \{0,1\}^m$ to be used by the reduction M, where t is a parameter chosen later, and m is the length of a coin flip used by M. The verifier sets $R := \{0,1\}^* \setminus I_0 \setminus I_1$, and accept if and only if $\Pr_{j,k\sim[t]}[M^R(x; r_j^0 \oplus r_k^1) = 1] > \frac{1}{2}$, where $M^R(x;r)$ denotes the output of the reduction when its coin flip is r, and $\oplus$ denotes the bit-wise XOR. Note that the running time of the verifier is at most a polynomial in n and t. Below we establish the correctness of this algorithm for some $t = \mathsf{poly}(n)$.

We focus on the case when the 0th prover is honest; thus $I_{0,\ell} := \mathrm{Im}(G_\ell)$ for each $\ell \leq c \log n$. Since $|I_{1,\ell}| \leq 2^{s(\ell)}$, the number of strings of length ℓ in $R(I_1) := \{0,1\}^* \setminus I_0 \setminus I_1$ is at least $2^\ell - |I_{0,\ell}| - |I_{1,\ell}| \geq 2^\ell \gamma(\ell)$ (here we write $R(I_1)$ instead of R to emphasize that R depends on I_1); thus $R(I_1)$ is a γ-avoiding oracle. By the definition of the reduction, for every I_1 we have $\Pr_{r\sim\{0,1\}^m}[M^{R(I_1)}(x;r) = L(x)] \geq \frac{2}{3}$.

We use the notion of cover introduced by Canetti [Can96]: A sequence $r_1, \cdots, r_t \in \{0,1\}^m$ is called a cover of a subset $A \subseteq \{0,1\}^m$ if for all $r \in \{0,1\}^m$, $\Pr_{j\sim[t]}[r_j \oplus r \in A] > \frac{1}{2}$. Define $A(I_1) := \{\, r \in \{0,1\}^m \mid M^{R(I_1)}(x;r) = L(x) \,\}$. We claim that by a probabilistic argument there exists a sequence $r_1, \cdots, r_t \in \{0,1\}^m$ that covers $A(I_1)$ for every I_1: Fix any I_1 and $r \in \{0,1\}^m$. Pick $r_1, \cdots, r_t \sim \{0,1\}^m$. For any $j \in [t]$, the probability that $r_j \oplus r \in A(I_1)$ is at least $\frac{2}{3}$. Thus by a concentration bound (Lemma 9), the probability that at most a $\frac{1}{2}$-fraction of $j \in [t]$ satisfies $r_j \oplus r \in A(I_1)$ is at most $\exp(-\Omega(t))$. By the union bound over all r, the probability that a sequence $r_1, \cdots r_t$ does not cover $A(I_1)$ is at most $2^m \cdot \exp(-\Omega(t))$. By the union bound over all I_1, the probability that there exists some I_1 such that $A(I_1)$ is not covered by $r_1, \cdots, r_t$ is at most $2^{n^c+m} \cdot \exp(-\Omega(t))$. Therefore, for $t := \Theta(n^c + m)$, there exists a sequence $r_1, \cdots, r_t$ that covers $A(I_1)$ for every I_1. The 0th honest prover sends this sequence $r_1, \cdots, r_t$ to the verifier as $r_1^0, \cdots, r_t^0$, in which case the verifier outputs $L(x)$ correctly because $\Pr_{j,k\sim[t]}[M^{R(I_1)}(x; r_j^0 \oplus r_k^1) = L(x)] > \frac{1}{2}$, for every I_1 and every $r_1^1, \cdots, r_t^1$. □

6 Simulating Long Queries by AM and coAM

In this section, we show that a reduction that makes only long queries to a distinguisher for a hitting set generator can be simulated in $\mathsf{AM} \cap \mathsf{coAM}$. Combining this $\mathsf{AM} \cap \mathsf{coAM}$-type simulation algorithm with Section 5, we will complete the proof of our main results.

Theorem 16 (Main; a restated version of Theorem 4)**.** *Let $G = \{G_\ell : \{0,1\}^{s(\ell)} \to \{0,1\}^\ell\}_{\ell \in \mathbb{N}}$ be any (not necessarily computable) family of functions and $\gamma\colon \mathbb{N} \to [0,1)$ be a parameter such that*

- *there exists a constant $\epsilon > 0$ such that $s(\ell) \leq (1-\epsilon)\ell$ for all large $\ell \in \mathbb{N}$, and*

[9] Alternatively, we may use the result of Russell and Sundaram [RS98] showing that $\mathsf{S}_2 \cdot \mathsf{BP} \cdot \mathsf{P} = \mathsf{S}_2^{\mathrm{p}}$ in a black-box way.

- *there exists a constant $c > 0$ such that $\gamma(\ell) \le 1 - \ell^{-c}$ for all large $\ell \in \mathbb{N}$.*

Then,

$$\bigcap_{R\colon \gamma\text{-avoids } G} \mathsf{BPP}_{\|}^{R} \quad \subseteq \quad \mathsf{NP/poly} \cap \mathsf{coNP/poly} \cap \mathsf{S}_2^{\mathsf{NP}}.$$

Moreover, if G can be computed in time $2^{O(\ell)}$, then we also have

$$\bigcap_{R\colon \gamma\text{-avoids } G} \mathsf{BPP}_{\|}^{R} \quad \subseteq \quad \mathsf{AM} \cap \mathsf{coAM}.$$

The advice in Theorem 16 is used in order to give the characteristic function of $\mathrm{Im}(G)$ for all strings of length $O(\log n)$, under which situation the rest of reductions can be simulated in $\mathsf{AM} \cap \mathsf{coAM} \subseteq \mathsf{NP/poly} \cap \mathsf{coNP/poly}$. If a hitting set generator is computable in exponential time, then the advice can be computed in polynomial time and thus can be removed. Without any advice and without any computational bound on a hitting set generator, the reduction can be simulated in $\mathsf{S}_2^{\mathsf{NP}}$, which is a complexity class that can simulate $\mathsf{AM} \cap \mathsf{coAM}$ and $\mathsf{S}_2^{\mathsf{p}}$.

The following is the main technical result of this section.

Theorem 17 (AM Simulation of Long Queries)**.** *Let $G = \{G_\ell : \{0,1\}^{s(\ell)} \to \{0,1\}^\ell\}_{\ell\in\mathbb{N}}$ be any family of functions. Let $t, \theta, \alpha_0 \colon \mathbb{N} \to \mathbb{N}$ be efficiently computable functions. Suppose that there exists a randomized $t(n)$-time nonadaptive black-box reduction M from a language L to any γ-avoiding oracle for G such that the length of any query that M makes on input length n is at least $\theta(n)$. Suppose also that, for all large $n \in \mathbb{N}$,*

- $\alpha_0(n) \le \frac{1}{16\mathrm{e}^3 t(n)^2} 2^{\theta(n) - s(\theta(n))}$, *and*
- $\alpha_0(n) \cdot \big(1 - 2^{s(\ell)-\ell} - \gamma(\ell)\big) \ge 1$ *for all $\ell \in \mathbb{N}$ such that $\theta(n) \le \ell \le t(n)$.*

Then, there exists an $\mathsf{AM} \cap \mathsf{coAM}$ protocol running in $t(n)^{O(1)}$ time that decides L.

We first observe that this is sufficient to prove Theorem 16.

Proof of Theorem 16 from Theorem 17. Take any language $L \in \bigcap_{R\colon \gamma\text{-avoids } G} \mathsf{BPP}_{\|}^{R}$. By Lemma 14, we have a randomized $t(n)$-time nonadaptive black-box reduction M from L to any γ-avoiding oracle for G, where $t(n) = n^{O(1)}$. By the assumption on the seed length s, we have $\epsilon\ell \le \ell - s(\ell)$ for all large $\ell \in \mathbb{N}$. For any $\ell \in \mathbb{N}$ between $\theta(n)$ and $t(n)$, we have $1 - 2^{s(\ell)-\ell} - \gamma(\ell) \ge \ell^{-c} - 2^{-\epsilon\ell} \gg \ell^{-c}/2$; hence, by defining $\alpha_0(n) := 2t(n)^c$, we obtain $\alpha_0(n) \ge \big(1 - 2^{s(\ell)-\ell} - \gamma(\ell)\big)^{-1}$ for all ℓ between $\theta(n) \le \ell \le t(n)$. On the other hand, $2^{\theta(n)-s(\theta(n))}/t(n)^2 \ge 2^{\epsilon\theta(n)}/t(n)^2$; thus, for $\theta(n) := ((c+2+1)\log t(n))/\epsilon$, we obtain $\alpha_0(n) \ll 2^{\theta(n)-s(\theta(n))}/16\mathrm{e}^3 t(n)^2$ for all large n.

The assumptions about parameters of Theorem 17 are thus satisfied for $\theta(n) = O(\log t(n))$. In particular, we can encode the characteristic function of the set $\cup_{\ell\le\theta(n)}\mathrm{Im}(G_\ell)$ as an advice string of length $t(n)^{O(1)}$. Given such an advice, we can modify the reduction M so that M does not make any query q of length at most $\theta(n)$: Indeed, if the original reduction makes a query q of length $\le \theta(n)$, then we modify the reduction so that q is answered according to whether $q \in \{0,1\}^* \setminus \mathrm{Im}(G_{|q|})$, which can be decided by using the advice. After this modification, by using Theorem 17, M can be simulated in $\mathsf{AM} \cap \mathsf{coAM}$. We thus obtain $L \in \mathsf{AM/poly} \cap \mathsf{coAM/poly} = \mathsf{NP/poly} \cap \mathsf{coNP/poly}$.

Moreover, if G is computable in $2^{O(\ell)}$, then the advice can be computed in polynomial time: Indeed, by an exhaustive search, one can compute $\mathrm{Im}(G_\ell)$ in time $2^{O(s(\ell))} \cdot 2^{O(\ell)} = 2^{O(\ell)} \leq t(n)^{O(1)}$ for all $\ell \leq \theta(n)$.

Finally, we sketch an $\mathsf{S}_2^{\mathsf{NP}}$ algorithm for deciding L when G is not necessarily computable and no advice is given: As in Theorem 15, each competing prover sends a verifier all the strings in $\mathrm{Im}(G)$ of length at most $\theta(n)$. Let $I_0, I_1 \subseteq \{0,1\}^*$ be the claimed image of G. Then we modify the reduction M so that any short query is answered according to $\{0,1\}^* \setminus I_0 \setminus I_1$. Now by applying Theorem 17, we obtain an AM algorithm deciding L. In particular, there exists an NP machine V such that $\Pr_r[V(x,r) = L(x)] \geq \frac{2}{3}$ for every input x. This randomized computation can be derandomized as in Theorem 15, by requesting each competing prover to send a sequence of coin flips $r_1, \cdots, r_s$. Thus we obtain $L \in \mathsf{S}_2^{\mathsf{NP}}$. □

The rest of this section is devoted to proving Theorem 17.

6.1 Generalized Feigenbaum–Fortnow Protocol

One of the main building blocks of our proof is a generalization of the protocol of Feigenbaum and Fortnow [FF93] (and its description by Bogdanov and Trevisan [BT06b]) for simulating some type of randomized nonadaptive reduction M to an NP problem R. Suppose that for a given input x, M makes m nonadaptive queries $q_1, \ldots, q_m$ under a certain distribution $\mathcal{Q}$. In the Feigenbaum–Fortnow protocol, a verifier asks a prover to give witnesses to all positive instances among them. The prover cannot give a witness to a negative instance (hence, it cannot cheat the verifier by saying "yes" to a negative instance) while it may try to cheat the verifier by not giving a witness to some of the positive instances of $q_1, \ldots, q_m$. If, however, the verifier knows the proportion p^* of positive instances among queries under the distribution $\mathcal{Q}$, then it may detect wrong negative answers from the prover if the number of positive answers is much smaller than p^*m. More specifically, the Feigenbaum–Fortnow protocol runs as follows. It first generates K tuples of m nonadaptive queries $\{(q_{k1}, \ldots, q_{km})\}_{1 \leq k \leq K}$ by running $M(x)$ independently K times. By a concentration inequality, the number of positive instances among all Km queries should be in the range of $m \cdot \left(p^*K \pm O(\sqrt{K})\right)$ with high probability; thus, if the prover gives "yes" answers (with witnesses) much smaller than $m \cdot \left(p^*K - O(\sqrt{K})\right)$, then the verifier stops the computation immediately, suspecting that the prover is not honest. On the other hand, if the number of positive answers to those queries is close to p^*Km, then the number of positive instances on which the prover can cheat is at most $O(m\sqrt{K})$, with high probability. We choose K large enough so that $O(m\sqrt{K}) \ll K$; then the majority of K tuples are answered correctly by the oracle, and we can use them to determine the result of $M^R(x)$ by taking the majority vote of the results of $M(x)$ computed by using prover's answers to each tuple of queries $(q_{k1}, \ldots, q_{km})$.[10]

We generalize the Feigenbaum–Fortnow protocol so that a new protocol is capable of dealing with a reduction to a *distributional* AM problem R; that is, we show that, given any nonadaptive reduction to some AM problem solvable on average and the proportion p^* of positive instances as advice, one can simulate the reduction in $\mathsf{AM} \cap \mathsf{coAM}$. In our protocol, we use Adleman's trick (for proving $\mathsf{BPP} \subseteq \mathsf{P/poly}$ [Adl78]) to "derandomize" AM oracle so that we obtain a new NP oracle, and then run the original Feigenbaum–Fortnow protocol. The following is the specification of the generalized Feigenbaum–Fortnow protocol:

[10] We note that taking the majority is not necessary; instead, it suffices to pick $k \sim [K]$ and use the result.

Inputs. A tuple (C, V, δ, p^*) such that:

- A randomized nonadaptive reduction C is given as a probabilistic circuit such that each query of C is identically distributed to some distribution $\mathcal{Q}$ over $\{0,1\}^*$, and the reduction always makes exactly m queries. (We assume that an input to a reduction is hardwired into the circuit C; thus C does not take any input other than random bits.)
- An AM verifier V is given as a circuit.
- An error parameter $\delta \in \left(0, \frac{1}{2}\right)$ is given in unary, and a probability $p^* \in [0,1]$ is given in binary.

Promise. We assume that there exist some answer $a \in \{0,1\}$, some oracle $R \subseteq \{0,1\}^*$, and some error parameters $\epsilon_0, \epsilon_1, \epsilon_2 \in [0,1]$ satisfying the following:

- $\Pr_C[C^R = a] \geq 1 - \epsilon_0$. (That is, a is supposed to be the answer of the reduction C to the oracle R.)
- The advice p^* satisfies $|p^* - \Pr_{q \sim \mathcal{Q}}[q \in R]| \leq \epsilon_1$.
- The distributional problem $(R, \mathcal{Q})$ is "solvable by AM on average": that is, define[11]

$$V_{\text{YES}} := \{\, q \in \{0,1\}^* \mid \Pr_r [V(q,y,r) = 1 \text{ for some } y] \geq 3/4 \,\}, \text{ and}$$
$$V_{\text{NO}} := \{\, q \in \{0,1\}^* \mid \Pr_r [V(q,y,r) = 0 \text{ for all } y] \geq 3/4 \,\};$$

  then we assume that $\Pr_{q \sim \mathcal{Q}}[q \in V_{\text{YES}} \cup V_{\text{NO}}] \geq 1 - \epsilon_2$ and $V_{\text{YES}} \subseteq R \subseteq \{0,1\}^* \setminus V_{\text{NO}}$.

Protocol.

1. (Preprocess of Verifier: Adleman's trick) Let s be sufficiently large so that $s > 20|V|$ and $s \geq 20 \log(1/\delta)$ where $|V|$ denotes the circuit size of V. Pick $r_1, \ldots r_s$ uniformly at random, and share the random bits with the prover. Define a new circuit W by

$$W(x, y_1, \ldots, y_s) := \operatorname*{majority}_{i \in [s]} V(x, y_i, r_i).$$

   In what follows, we call $\bar{y} := (y_1, \ldots, y_s)$ a certificate for W.

2. (Verifier) Let $K := m^2 (1/\delta)^2 \log(m/\delta)$. Run C independently K times and obtain queries $(q_{k1}, \ldots, q_{km})$ for each kth run of C ($k \in [K]$). Send these queries to the prover.

3. (Prover) For each $(k,i) \in [K] \times [m]$, send a certificate $\bar{y}_{ki}$ for W; an honest prover sends, if any, some certificate $\bar{y}_{ki}$ such that $W(q_{ki}, \bar{y}_{ki}) = 1$.

4. (Verifier) Let $a^*_{ki} := W(q_{ki}, \bar{y}_{ki}) \in \{0,1\}$ for each $(k,i) \in [K] \times [m]$. Verify that

$$\sum_{\substack{1 \leq k \leq K \\ 1 \leq i \leq m}} a^*_{ki} \;\geq\; mp^*K \;-\; m\left((\epsilon_1 + \epsilon_2)K + \sqrt{K \log(m/\delta)}\right), \tag{2}$$

   and if not, output $\perp$ and halt. Otherwise, pick $k \sim [K]$ uniformly at random and output the kth run of the reduction of C assuming that the answers from the oracle are $(a_{k1}, \ldots, a_{km})$.

[11] As a circuit V outputs "undefined" if an input (q,y,r) is too long, the sets $V_{\text{YES}}, V_{\text{NO}}$ of strings are finite.

Theorem 18 (Correctness of the Generalized Feigenbaum–Fortnow Protocol)**.** *Suppose that the protocol above is given inputs satisfying the promise listed above. Then, the protocol satisfies the completeness and soundness for error* $\epsilon := \epsilon_0 + 2m\epsilon_1 + 3m\epsilon_2 + 3\delta$*, described below:*

- *(Completeness) There exists a prover such that the verifier outputs* a *with probability at least* $1 - \epsilon$*.*
- *(Soundness) For any prover, the verifier outputs* a *or* $\perp$ *with probability at least* $1 - \epsilon$*.*

We prove this theorem by a sequence of claims below. The following claim follows from a standard fact about amplification of the success probability of a randomized machine.

Claim 19 (Amplification and Adleman's trick)**.** *With probability at least* $1 - \delta$ *over the choice of* $r_1, \ldots, r_s$*, the following holds. For any query* $q \in V_{\mathrm{YES}} \cup V_{\mathrm{NO}}$*,*

- *if* $q \in V_{\mathrm{YES}}$ *then* $W(q, \bar{y}) = 1$ *for some certificate* $\bar{y} := (y_1, \ldots, y_s)$*, and*
- *if* $q \in V_{\mathrm{NO}}$ *then* $W(q, \bar{y}) = 0$ *for any certificate* $\bar{y} := (y_1, \ldots, y_s)$*.*

Proof. First, note that $|V_{\mathrm{YES}} \cup V_{\mathrm{NO}}|$ is at most $2^{s/20}$. Indeed, the circuit size of V is less than $s/20$, and hence for any input q of length $\geq s/20$, V is not defined; thus $q \notin V_{\mathrm{YES}} \cup V_{\mathrm{NO}}$. That is, the length of every query in $V_{\mathrm{YES}} \cup V_{\mathrm{NO}}$ is less than $s/20$.

Fix any q such that $q \in V_{\mathrm{YES}}$ or $q \in V_{\mathrm{NO}}$; in the former case, let $a_q := 1$ and $a_q := 0$ otherwise. Our claim is that $\max_{\bar{y}} W(q, \bar{y}) = a_q$. For each $i \in [s]$, let $X_i \in \{0, 1\}$ be the random variable (over the random choice of $r_1, \ldots, r_s$) such that $X_i := 1$ iff $V(q, y_i, r_i) = 1$ for some y_i; in other words, $X_i := \max_{y_i} V(q, y_i, r_i) \in \{0, 1\}$. Observe that

$$\max_{\bar{y}} W(q, \bar{y}) = \max_{\bar{y}} \operatorname*{majority}_{i \in [s]} V(q, y_i, r_i) = \operatorname*{majority}_{i \in [s]} \max_{y_i} V(q, y_i, r_i) = \operatorname*{majority}_{i \in [s]} X_i.$$

By the assumption on V, we have $|\mathbb{E}[X_i] - a_q| \leq \frac{1}{4}$ for any $i \in [s]$; hence, $\operatorname{majority}_{i \in [s]} X_i \neq a_q$ implies $|\frac{1}{s} \sum_i (X_i - \mathbb{E}[X_i])| \geq \frac{1}{4}$. By Hoeffding's inequality (Lemma 9),

$$\Pr[\max_{\bar{y}} W(q, \bar{y}) \neq a_q] \leq \Pr\left[\left|\sum_{i=1}^{s} (X_i - \mathbb{E}[X_i])\right| \geq \frac{s}{4}\right] \leq 2 \exp\left(-2s\,(1/4)^2\right) \leq 2^{-s/10}.$$

Now, by the union bound over all $q \in V_{\mathrm{YES}} \cup V_{\mathrm{NO}}$, the probability that there exists some $q \in V_{\mathrm{YES}} \cup V_{\mathrm{NO}}$ such that $\max_{\bar{y}} W(q, \bar{y}) \neq a_q$ is at most $|V_{\mathrm{YES}} \cup V_{\mathrm{NO}}| \cdot 2^{-s/10} \leq 2^{-s/20} \leq \delta$. □

For each $(k, i) \in [K] \times [m]$, define $a_{ki} \in \{0, 1\}$ as $a_{ki} := 1$ if and only if $W(q_{ki}, \bar{y}) = 1$ for some certificate $\bar{y}$. The honest prover sends a certificate for W (if any), and thus $a_{ki} = a^*_{ki}$; on the other hand, when communicating with a cheating prover, we have only $a^*_{ki} \leq a_{ki}$. The next claim shows the sum of (a_{ki}) concentrates around its mean.

Claim 20 (Concentration)**.** *Under the event of Claim 19, with probability at least* $1 - \delta$*, the following holds:*

$$\left|\sum_{\substack{1 \leq k \leq K \\ 1 \leq i \leq m}} a_{ki} - mKp^*\right| \leq m\left((\epsilon_1 + \epsilon_2)K + \sqrt{K \log(m/\delta)}\right).$$

Proof. Fix any $i \in [m]$. By the assumption on C, the queries $q_{1i}, \ldots, q_{Ki}$ are independent and identically distributed according to $\mathcal{Q}$; hence, $a_{1i}, \ldots, a_{Ki} \in \{0,1\}$ are also independent random variables. The expectation $\mathbb{E}[a_{ki}]$ of these random variables is equal to $\Pr_{q\sim\mathcal{Q}}[W(q,\bar{y}) = 1 \text{ for some } \bar{y}]$.

We claim that, for any $(k,i) \in [K] \times [m]$, the expectation $\mathbb{E}[a_{ki}]$ is equal to p^* up to additive error $\epsilon_1 + \epsilon_2$. Under the event of Claim 19, we have $q \in V_{\text{YES}} \implies \exists \bar{y}.\ W(q,\bar{y}) = 1 \implies q \notin V_{\text{NO}}$ for any $q \in \{0,1\}^*$; hence,

$$\Pr[q \in V_{\text{YES}}] \le \Pr[\exists \bar{y}.\ W(q,\bar{y}) = 1] \le \Pr[q \notin V_{\text{NO}}]. \tag{3}$$

Similarly, since $V_{\text{YES}} \subseteq R \subseteq \{0,1\}^* \setminus V_{\text{NO}}$, we have

$$\Pr[q \in V_{\text{YES}}] \le \Pr[q \in R] \le \Pr[q \notin V_{\text{NO}}]. \tag{4}$$

Combining (3) and (4), we obtain

$$|\Pr[q \in R] - \mathbb{E}[a_{ki}]| \le \Pr[q \notin V_{\text{NO}}] - \Pr[q \in V_{\text{YES}}] \le \epsilon_2.$$

Therefore, $|\mathbb{E}[a_{ki}] - p^*| \le |\mathbb{E}[a_{ki}] - \Pr[q \in R]| + |\Pr[q \in R] - p^*| \le \epsilon_2 + \epsilon_1$.

By Hoeffding's inequality (Lemma 9), for each $i \in [m]$,

$$\Pr\left[\left|\sum_{k=1}^{K}(a_{ki} - \mathbb{E}[a_{ki}])\right| \ge \sqrt{K\log(m/\delta)}\right] \le 2\exp(-2K\log(m/\delta)/K) \le \delta/m.$$

By the union bound over all $i \in [m]$, with probability at least $1-\delta$, we have

$$\begin{aligned}\left|\sum_{i=1}^{m}\sum_{k=1}^{K} a_{ki} - mKp^*\right| &\le \left|\sum_{i=1}^{m}\sum_{k=1}^{K}(a_{ki} - \mathbb{E}[a_{ki}])\right| + \left|\sum_{i=1}^{m}\sum_{k=1}^{K}(\mathbb{E}[a_{ki}] - p^*)\right| \\ &\le \sum_{i=1}^{m}\left|\sum_{k=1}^{K}(a_{ki} - \mathbb{E}[a_{ki}])\right| + mK(\epsilon_1+\epsilon_2) \\ &\le m\sqrt{K\log(m/\delta)} + mK(\epsilon_1+\epsilon_2).\end{aligned}$$

□

Now we are ready to bound the probability of completeness and soundness. Let E denote any event (which is supposed to be the event that completeness or soundness does not hold); using Claim 19 and 20, we will bound the probability in the following way:

$$\Pr[E] \ \le\ 2\delta + \Pr[\,E \wedge (\text{the event of Claim 19 holds}) \wedge (\text{the concentration of Claim 20 occurs})\,].$$

That is, assuming that the events of Claim 19 and 20 happens, we will analyze the probability of completeness and soundness.

Claim 21 (Completeness)**.** *The verifier outputs a with probability at least $1-\epsilon$ when interacting with the honest prover.*

Proof. The verifier *does not* output a only if

- the inequality (2) does not hold, or
- for a random $k \sim [K]$, the kth run of C is not correct.

The honest prover sets $a^*_{ki} := a_{ki}$ for any $(k,i) \in [K] \times [m]$. Thus, under the assumption that the concentration of Claim 20, the inequality (2) is satisfied; that is, the verifier does not output $\perp$, and hence it remains to bound the probability that, for a random $k \sim [K]$, the kth run of C is not correct.

The kth run of C is not correct only if the reduction itself makes a mistake, or there exists some $i \in [m]$ such that $a^*_{ki} \neq R(q_{ki})$ (which happens only if $q_{ki} \notin V_{\mathrm{YES}} \cup V_{\mathrm{NO}}$ for the honest prover). The former probability is at most ϵ_0, and the latter is at most $m\epsilon_2$.

Overall, the verifier outputs a with probability at least $1 - 2\delta - \epsilon_0 - m\epsilon_2 \geq 1 - \epsilon$. □

Claim 22 (Soundness)**.** *For any cheating prover, the verifier outputs a or $\perp$ with probability at least $1 - \epsilon$.*

Proof. The verifier outputs the wrong answer $1 - a$ only if for a random $k \sim [K]$, the kth run of C is not correct.

Recall that we have $a^*_{ki} \leq a_{ki}$ for any $(k,i) \in [K] \times [m]$ no matter how a prover tries to cheat. The main difference between the proof of Claim 21 is that, for a random choice $k \sim [K]$ of the verifier, a prover may be cheating so that $a^*_{ki} < a_{ki}$ for some $i \in [m]$; as a consequence, the kth run of C is more likely to be wrong. On the other hand, the number of $(k,i) \in [K] \times [m]$ such that $a^*_{ki} < a_{ki}$ is small: Indeed, under the event that the verifier does not output $\perp$, the inequality (2) holds, and we also have the concentration of Claim 20; hence, we obtain $\sum_{k=1}^{K} \sum_{i=1}^{m} (a_{ki} - a^*_{ki}) \leq 2m \left((\epsilon_1 + \epsilon_2)K + \sqrt{K \log(m/\delta)} \right)$. Thus, the probability that $a^*_{ki} < a_{ki}$ for some $i \in [m]$ over the random choice of $k \sim [K]$ is at most $2m(\epsilon_1 + \epsilon_2) + m\sqrt{\log(m/\delta)/K} \leq 2m(\epsilon_1 + \epsilon_2) + \delta$.

Overall, the probability that the verifier outputs the wrong answer is at most $(2\delta + \epsilon_0 + m\epsilon_2) + (2m(\epsilon_1 + \epsilon_2) + \delta) \leq \epsilon$. □

Proof of Theorem 18. Immediate from Claim 21 and 22. □

6.2 Simulating Long Queries

Using the generalized Feigenbaum–Fortnow protocol, we show how to simulate long queries by a constant-round interactive proof system (i.e., a proof of Theorem 17). For simplicity, we focus on the case when $t(n) = n^{O(1)}$. Let M be a randomized $t(n)$-time nonadaptive black-box reduction to any γ-avoiding oracle for G. Let $x \in \{0,1\}^*$ be an input of length n.

We first modify the reduction so that we can assume useful properties. By the modifications explained in Section 4, we may assume that the number of queries that M makes is exactly $m(n)$ $(\leq t(n))$ on inputs of length n. We may also assume that each query of M is identically distributed; Let $\mathcal{Q}_x$ be the query distribution of M on input x.

As explained in the introduction, one of the keys of our proof is that we can replace a γ-avoiding oracle for G by an oracle defined based only on the query distribution $\mathcal{Q}_x$. Here we introduce such

oracles and justify the replacement. For any $\alpha > 0$, define L_α, H_α and R_α by

$$\begin{aligned} L_\alpha &:= \{\, q \in \{0,1\}^* \mid q \text{ is } \alpha\text{-light with respect to } \mathcal{Q}_x \,\}, \\ H_\alpha &:= \{0,1\}^* \setminus L_\alpha \;=\; \{\, q \in \{0,1\}^* \mid q \text{ is } \alpha\text{-heavy with respect to } \mathcal{Q}_x \,\}, \text{ and} \\ R_\alpha &:= L_\alpha \setminus \mathrm{Im}(G). \end{aligned}$$

For large enough $\alpha > 0$, we can easily show that R_α γ-avoids G.

Claim 23. *For any $\gamma\colon \mathbb{N} \to [0,1)$ and $\alpha > 0$ and for any length $\ell \in \mathbb{N}$, if*

$$\gamma(\ell) + 1/\alpha \le 1 - 2^{s(\ell)-\ell},$$

then R_α is a γ-avoiding set at length ℓ for G.

Proof. Since $R_\alpha \subseteq \{0,1\}^* \setminus \mathrm{Im}(G)$, it suffices to show that $\Pr_{w\sim\{0,1\}^\ell}[w \notin R_\alpha] \le 1 - \gamma(\ell)$. Note that $w \notin R_\alpha$ if either $w \in \mathrm{Im}(G_\ell)$ or w is α-heavy. The probability of the former case is at most $2^{-\ell} \cdot |\mathrm{Im}(G_\ell)| \le 2^{s(\ell)-\ell}$. Similarly, the probability of the latter case is bounded above by $2^{-\ell} \cdot |H_\alpha^{=\ell}|$, where $H_\alpha^{=\ell} = \{\, q \in \{0,1\}^\ell \mid q \text{ is } \alpha\text{-heavy} \,\}$. On the other hand, we have

$$|H_\alpha^{=\ell}| \cdot \alpha 2^{-\ell} \;\le\; \sum_{q \in H_\alpha^{=\ell}} \Pr_{w\sim\mathcal{Q}_x}[\, w = q \,] \;=\; \Pr_{w\sim\mathcal{Q}_x}[\, w \in H_\alpha^{=\ell} \,] \;\le\; 1.$$

Hence, $|H_\alpha^{=\ell}| \le 2^\ell/\alpha$. Thus,

$$\Pr_{w\sim\{0,1\}^\ell}[w \notin R_\alpha] \;\le\; 2^{s(\ell)-\ell} + 1/\alpha \;\le\; 1 - \gamma(\ell),$$

proving that $\Pr_{w\sim\{0,1\}^\ell}[w \in R_\alpha] \ge \gamma(\ell)$. □

Since the reduction M does not make any query q such that $|q| \notin [\theta(n), t(n)]$, Claim 23 guarantees that the reduction M works by using R_α on inputs x of length n if $\gamma(\ell) + 1/\alpha \le 1 - 2^{s(\ell)-\ell}$ for all $\ell \in \mathbb{N}$ such that $\theta(n) \le \ell \le t(n)$. As this condition is satisfied by our assumptions of Theorem 17 for any $\alpha \ge \alpha_0(n)$ and any input x of length n, we have

$$\Pr_M\left[M^{R_\alpha}(x) = L(x)\right] \;\ge\; \frac{15}{16}. \tag{5}$$

On the other hand, we can show below that $M(x)$ cannot distinguish R_α and L_α when α is small enough.

Claim 24. *For any $\alpha > 0$ and input $x \in \{0,1\}^*$ of length n, and for $\epsilon := \alpha 2^{s(\theta(n))-\theta(n)} \cdot m(n)t(n)$,*

$$\Pr_M\left[\, M^{L_\alpha}(x) \neq M^{R_\alpha}(x) \,\right] \;\le\; \epsilon. \tag{6}$$

Proof. Recall that $R_\alpha = L_\alpha \setminus \mathrm{Im}(G)$. Thus, $M(x)$ may find the difference between L_α and R_α only if it makes a query in $L_\alpha \cap \mathrm{Im}(G)$ in one of its $m(n)$ nonadaptive queries. This probability is at most $m(n) \cdot \Pr_{w\sim\mathcal{Q}_x}[w \in L_\alpha \cap \mathrm{Im}(G)]$ by a union bound.

Here we have

$$\begin{aligned}\Pr_{w\sim\mathcal{Q}_x}[\,w\in L_\alpha\cap\mathrm{Im}(G)\,] &= \sum_{q\in L_\alpha\cap\mathrm{Im}(G)}\Pr_{w\sim\mathcal{Q}_x}[\,q=w\,]\\ &\le \sum_{q\in\mathsf{supp}(\mathcal{Q}_x)\cap\mathrm{Im}(G)}\alpha\cdot 2^{-|q|},\end{aligned}$$

where $\mathsf{supp}(\mathcal{Q}_x)$ is the set of all possible queries asked by $M(x)$. By our assumption on M, we have $\theta(n)\le|q|\le t(n)$ for any $q\in\mathsf{supp}(\mathcal{Q}_x)$. Then it follows

$$\sum_{q\in\mathsf{supp}(\mathcal{Q}_x)\cap\mathrm{Im}(G)}\alpha\cdot 2^{-|q|} \le \sum_{\theta(n)\le\ell\le t(n)}\alpha\cdot 2^{-\ell}\cdot|\mathrm{Im}(G_\ell)| \le \sum_{\theta(n)\le\ell\le t(n)}\alpha\cdot 2^{s(\ell)-\ell}$$

because $|\mathrm{Im}(G_\ell)|\le 2^{s(\ell)}$.

Since we assumed that $\ell-s(\ell)$ is nondecreasing for $\ell\in\mathbb{N}$, we have

$$\sum_{\theta(n)\le\ell\le t(n)}\alpha 2^{s(\ell)-\ell} \le t(n)\cdot\alpha 2^{s(\theta(n))-\theta(n)} = \epsilon/m(n).$$

This bound is sufficient to get the desired error bound. □

By Claim 24 and our assumptions on $\alpha_0(n)$ of Theorem 17, for any $\alpha\le \mathrm{e}^3\alpha_0(n)$, we have

$$\Pr_M\left[\,M^{L_\alpha}(x)\ne M^{R_\alpha}(x)\,\right]\le\frac{1}{16}. \tag{7}$$

From the inequalities (5) and (7), we immediately obtain the following:

Corollary 25. *For any input $x\in\{0,1\}^*$ of length n and any $\alpha\in[\alpha_0(n),\mathrm{e}^3\alpha_0(n)]$,*

$$\Pr_M\left[M^{L_\alpha}(x)=L(x)\right] \ge \frac{7}{8}.$$

In light of this, our task is now to simulate $M^{L_\alpha}(x)$ for some $\alpha\in[\alpha_0(n),\mathrm{e}^3\alpha_0(n)]$ (in fact, we will choose the threshold α randomly, as explained later). To this end, we combine the generalized Feigenbaum–Fortnow protocol, the lower bound protocol of Goldwasser and Sipser [GS86], and the heavy-sample protocol of Bogdanov and Trevisan [BT06b]. Here we review the last two protocols. Since these protocols are explained carefully and in detail in the paper [BT06b], we simply review their specifications and use them as a black-box tool.

6.2.1 Lower Bound Protocol

Recall that $q\notin L_\alpha$ if and only if q is α-heavy. The lower bound protocol of Goldwasser and Sipser [GS86] can be used to give an AM-type witness to any α-heavy instance. It is an AM protocol for showing that a given set of strings has more than s elements for a given threshold s. The specification of the lower bound protocol is as follows.

Inputs. A set of strings is given as a circuit C on $\{0,1\}^m$, which specifies the set as $C^{-1}(1):=\{\,r\in\{0,1\}^m\mid C(r)=1\,\}$. A threshold $s\in\mathbb{N}$ such that $0\le s\le 2^m$. Parameters $\delta,\epsilon\in[0,1]$ represented in unary.

Promise.

- Yes instances: $|C^{-1}(1)| \geq s$.
- No instances: $|C^{-1}(1)| \leq (1-\epsilon)s$.

Sketch of the Protocol.

1. (Verifier) Send a random hash function $h : \{0,1\}^m \to \{0,1\}^{m'}$ for some appropriate parameter m'.
2. (Prover) Send a string $y \in \{0,1\}^m$ claiming that $y \in C^{-1}(1)$ and $h(y) = 0^{m'}$ hold. (Such an y is called *an* AM*-type witness.*)
3. (Verifier) Check the correctness of the prover's claim on the witness y.

Theorem 26 (Correctness of the lower bound protocol [GS86]; see [BT06b] for a proof)**.** *The lower bound protocol stated above satisfies the following:*

- *(Completeness) Given an yes instance, there exists a prover that makes the verifier accept with probability at least* $1-\delta$.
- *(Soundness) Given a no instance, for any prover, the verifier accepts with probability at most* δ.

By using the protocol above, it is easy to construct an AM verifier V that checks whether a given query q is α-heavy.

Claim 27. *For any parameter* $\epsilon(n) \geq 1/\mathsf{poly}(n)$*, there exists an* AM *verifier* V *such that, for any input* $x \in \{0,1\}^*$ *of length* n *and any query* $q \in \{0,1\}^*$*,*

1. *if* q *is* α*-heavy with respect to* $\mathcal{Q}_x$*, then* $\Pr_h[V(x,q,h,y) = 1 \text{ for some } y] \geq \frac{3}{4}$*, and*
2. *if* q *is* $(1-\epsilon(n))\alpha$*-light with respect to* $\mathcal{Q}_x$*, then* $\Pr_h[V(x,q,h,y) = 1 \text{ for some } y] \leq \frac{1}{4}$.

Proof. Let Q_x be the circuit that samples the query distribution $\mathcal{Q}_x$; that is, on input $r \in \{0,1\}^m$, the circuit $Q_x(r)$ outputs q so that $\Pr_{r\sim\{0,1\}^m}[Q_x(r) = q] = \Pr_{w\sim\mathcal{Q}_x}[w = q]$ for any $q \in \{0,1\}^*$. Given a string $q \in \{0,1\}^*$ as input, construct a circuit C_q such that $C_q(r) := 1$ iff $Q_x(r) = q$, on input $r \in \{0,1\}^m$. Now use the lower bound protocol for the circuit C_q, the threshold $s := \alpha 2^{-|q|}2^m$, and parameters $\delta := \frac{1}{4}$ and $\epsilon := \epsilon(n)$. The lower bound protocol gives an AM certificate for the yes instances such that $|C_q^{-1}(1)| \geq s$, which is equivalent to saying that $\Pr_r[Q_x(r) = q] \geq \alpha 2^{-|q|}$, that is, q is α-heavy. On the other hand, if q is $(1-\epsilon)\alpha$-light, then we have $|C_q^{-1}(1)| < (1-\epsilon)s$; thus with high probability there is no AM-type witness by the correctness of the lower bound protocol (Theorem 26). □

Note that there is a gap between yes instances and no instances; that is, if the probability that q is sampled from $\mathcal{Q}_x$ is between α and $(1-\epsilon)\alpha$, then the behavior of the lower bound protocol is undefined. To circumvent this, we pick the threshold α randomly in the same way with Bogdanov and Trevisan (cf. [BT06b, Claim 3.2]): Consider the following set $\mathcal{A}_{\alpha_0,\epsilon}$ of thresholds defined by parameters $\alpha_0, \epsilon > 0$, and choose the threshold α uniformly at random from $\mathcal{A}_{\alpha_0,\epsilon}$.

$$\mathcal{A}_{\alpha_0,\epsilon} := \{\, \alpha_0(1+3\epsilon)^i \mid 0 \leq i \leq 1/\epsilon \,\}.$$

Observe that $\mathcal{A}_{\alpha_0,\epsilon} \subseteq [\alpha_0, \mathrm{e}^3\alpha_0]$. Moreover, the following holds.

Lemma 28. *For every $\alpha_0 > 0$ and $0 < \epsilon < \frac{1}{3}$ and any constant $c > 0$, and for any distribution $\mathcal{Q}$, with probability at least $1 - 1/c$ over the choice of $\alpha \sim \mathcal{A}_{\alpha_0,\epsilon}$,*

$$\Pr_{q\sim\mathcal{Q}}\left[\mathcal{Q}(q) \in ((1-\epsilon)\alpha 2^{-|q|}, (1+\epsilon)\alpha 2^{-|q|})\right] \leq c\epsilon. \tag{8}$$

(Recall that $\mathcal{Q}(q) := \Pr_{w\sim\mathcal{Q}}[w = q]$.)

Proof. For any $\epsilon \in \left(0, \frac{1}{3}\right)$ and $q \in \{0,1\}^*$, the intervals $((1-\epsilon)\alpha 2^{-|q|}, (1+\epsilon)\alpha 2^{-|q|})$ are pairwise disjoint for all $\alpha \in \mathcal{A}_{\alpha_0,\epsilon}$; hence for any real $p \in \mathbb{R}$, the probability that $p \in ((1-\epsilon)\alpha 2^{-|q|}, (1+\epsilon)\alpha 2^{-|q|})$ is at most $1/|\mathcal{A}_{\alpha_0,\epsilon}| \leq \epsilon$ over the choice of $\alpha \sim \mathcal{A}_{\alpha_0,\epsilon}$. In particular, we have

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\alpha\sim\mathcal{A}_{\alpha_0,\epsilon}}\left[\Pr_{q\sim\mathcal{Q}}\left[\mathcal{Q}(q) \in ((1-\epsilon)\alpha 2^{-|q|}, (1+\epsilon)\alpha 2^{-|q|})\right]\right] \\
&= \mathop{\mathbb{E}}_{q\sim\mathcal{Q}}\left[\Pr_{\alpha\sim\mathcal{A}_{\alpha_0,\epsilon}}\left[\mathcal{Q}(q) \in ((1-\epsilon)\alpha 2^{-|q|}, (1+\epsilon)\alpha 2^{-|q|})\right]\right] \;\leq\; \epsilon.
\end{aligned}$$

Therefore, by Markov's inequality, the probability that $\Pr_{q\sim\mathcal{Q}}\left[\mathcal{Q}(q) \in ((1\pm\epsilon)\alpha 2^{-|q|})\right] \geq c\epsilon$ is at most $\epsilon/(c\epsilon) = 1/c$. □

In our simulation protocol for M, we start with picking $\alpha \sim \mathcal{A}_{\alpha_0,\epsilon}$ randomly. By Lemma 28, except for probability $1/O(1)$, the heaviness of almost all queries q sampled from $\mathcal{Q}_x$ is not close to the threshold α. As a consequence, the distributional problem $(L_\alpha, \mathcal{Q}_x)$ is solvable by coAM on average; indeed, with probability at least $1 - O(\epsilon)$ over the choice of $q \sim \mathcal{Q}_x$, the lower bound protocol of Claim 27 solves L_α.

6.2.2 Heavy-sample Protocol

Next we review the heavy-sample protocol of Bogdanov and Trevisan [BT06b], which is an AM protocol for estimating $\Pr_{q\sim\mathcal{Q}_x}[q \text{ is } \alpha\text{-heavy}]$.

Inputs. A circuit Q which samples a string according to a distribution $\mathcal{Q}$ on $\{0,1\}^*$. A probability $p \in [0,1]$ represented in binary. Parameters $c > 0$ and $0 < \epsilon < \frac{1}{3}$ represented in unary. A threshold $\alpha > 0$ represented in binary.

Promise.

- Yes instances: $\Pr_{q\sim\mathcal{Q}}[\mathcal{Q}(q) \geq \alpha 2^{-|q|}] = p$.
- No instances: $\left|\Pr_{q\sim\mathcal{Q}}[\mathcal{Q}(q) \geq \alpha 2^{-|q|}] - p\right| > 16c\epsilon$.
- We assume the condition (8). That is,

$$\Pr_{q\sim\mathcal{Q}}\left[\mathcal{Q}(q) \in ((1-\epsilon)\alpha 2^{-|q|}, (1+\epsilon)\alpha 2^{-|q|})\right] \leq c\epsilon.$$

Sketch of the Protocol.

1. (Verifier) Generate random queries $q_1, \ldots, q_k$ from the distribution $\mathcal{Q}$ for some sufficiently large k, and send these queries to the prover.

2. (Prover) For each query q_i, tell the verifier whether q_i is α-heavy.

3. (Verifier and Prover) To check the prover's claim, run the lower bound protocol of Goldwasser and Sipser [GS86] and the upper bound protocol of Fortnow [For89] in parallel.

Theorem 29 (Correctness of the heavy-sample protocol [BT06b])**.** *The heavy-sample protocol specified above satisfies following:*

- *(Completeness) Given any yes instance, there exists a prover that makes the verifier accept with probability at least* $1 - O(\epsilon)$*.*
- *(Soundness) Given any no instance, for any prover, the verifier accepts with probability at most* $O(\epsilon)$*.*

6.2.3 Putting It Together

Using the protocols reviewed above, we can now simulate the reduction M to an oracle L_α on input $x \in \{0,1\}^*$. Below we explain how to choose the inputs $(C, V, \delta := \frac{1}{100}, p^*)$ for the generalized Feigenbaum–Fortnow protocol.

The generalized Feigenbaum–Fortnow protocol requires an AM protocol for solving an oracle on average (instead of coAM). By negating answers from the oracle, we can define a new machine $\overline{M}^X$ which simulates the computation of $M^{\{0,1\}^* \setminus X}$ for any given oracle X. We thus use the generalized Feigenbaum–Fortnow protocol for simulating $\overline{M}^{H_\alpha}(x)$ with oracle $H_\alpha := \{0,1\}^* \setminus L_\alpha$, which is the set of α-heavy queries with respect to $\mathcal{Q}_x$; more specifically, let C be the circuit that simulates the reduction $\overline{M}$ on input x (where the input x is hardwired into the circuit) and we give the circuit C to the protocol as input.

To solve H_α on average by an AM protocol, we use the lower bound protocol. That is, we build a circuit V_x that simulates the AM verifier stated in Claim 27 on input x and on all the queries $q \in \{0,1\}^*$ that $M(x)$ can make. Then we give the circuit V_x to the protocol as input.

We also need to give as advice a probability p^* that approximates $\Pr_{q \sim \mathcal{Q}_x}[q \in H_\alpha]$, which can be estimated by using the heavy-sample protocol. We require the prover to send p^* first, and then we verify the prover's claim by running the heavy-sample protocol; if the test passes, then we give p^* to the generalized Feigenbaum–Fortnow protocol as input.

Summarizing the discussion above, our whole simulation algorithm is given below.

Inputs. A string $x \in \{0,1\}^*$ of length n.

Promise. Let $\alpha_0 := \alpha_0(n)$. Then, for any $\alpha \in [\alpha_0, \mathrm{e}^3\alpha_0]$, we assume that

$$\Pr_M[M^{L_\alpha}(x) = L(x)] \geq \frac{7}{8},$$

(which is guaranteed by Corollary 25).

Protocol.

1. (Preprocess) Set an error parameter $\epsilon := 1/c_0 m(n)$ for a sufficiently large constant c_0 (represented in unary).

2. (Verifier) Pick a threshold $\alpha \sim \mathcal{A}_{\alpha_0,\epsilon} \subseteq [\alpha_0, \mathrm{e}^3\alpha_0]$ randomly. Send α to the prover.

3. (Prover) Send $p^* \in [0,1]$ to the verifier. An honest prover sends $p^* := \Pr_{q\sim\mathcal{Q}_x}[q \in H_\alpha]$.

4. (Verifier and Prover) Run the heavy-sample protocol in order to verify that $p^* \approx \Pr_{q\sim\mathcal{Q}_x}[q \in H_\alpha]$ for the distribution $\mathcal{Q}_x$ and error parameter ϵ and parameter $c := 100$; if the test does not pass, output $\perp$ and halt.

5. (Verifier and Prover) Build a circuit C simulating $\overline{M}$ on the hardwired input x, and a circuit V_x simulates the AM verifier for α-heaviness. Run the generalized Feigenbaum–Fortnow protocol on input $(C, V_x, \delta := \frac{1}{100}, p^*)$, and output the result of the protocol.

Now we argue that our simulation protocol is correct:

Claim 30. *The simulation protocol stated above satisfies the following: for any $x \in \{0,1\}^*$,*

- *(Completeness) there exists a prover such that the verifier outputs $L(x)$ with probability at least $3/4$, and*
- *(Soundness) for any prover, the verifier outputs $L(x)$ or $\perp$ with probability at least $3/4$.*

Proof. Fix any input $x \in \{0,1\}^*$. By Lemma 28, with probability at least $1 - \frac{1}{100}$ over the choice of $\alpha \sim \mathcal{A}_{\alpha_0,\epsilon}$, the condition (8) holds for $c := 100$ and $\mathcal{Q} := \mathcal{Q}_x$; that is,

$$\Pr_{q\sim\mathcal{Q}_x}\left[\mathcal{Q}_x(q) \in ((1-\epsilon)\alpha 2^{-|q|}, (1+\epsilon)\alpha 2^{-|q|})\right] \leq 100\epsilon.$$

In what follows, we assume this event happens and analyze the probability of the completeness and soundness.

Suppose that a prover sends p^*. If the prover is honest then we have $p^* = \Pr_{q\sim\mathcal{Q}_x}[q \in H_\alpha]$. Hence the completeness of the heavy-sample protocol implies that, with probability at least $1 - O(\epsilon) \gg 1 - \frac{1}{100}$, the protocol accepts. On the other hand, if a cheating prover sends p^* such that $|p^* - \Pr_{q\sim\mathcal{Q}_x}[q \in H_\alpha]| > 16c\epsilon$, then with probability at least 0.99 the cheat can be caught by the soundness of the heavy-sample protocol.

Thus, at the point that the generalized Feigenbaum–Fortnow protocol starts, for any prover, with probability at least 0.98, we have $|p^* - \Pr_{q\sim\mathcal{Q}_x}[q \in H_\alpha]| \leq 16c\epsilon$ and moreover the condition (8) holds. Under this event, the promise of the generalized Feigenbaum–Fortnow protocol is satisfied:

- Define $a := L(x)$, $\epsilon_0 := \frac{1}{8}$, and $R := H_\alpha$. Then we have $\Pr_C[C^R = a] \geq 1 - \epsilon_0$ by the promise of our simulation protocol (Corollary 25).
- The advice p^* satisfies $|p^* - \Pr_{q\sim\mathcal{Q}_x}[q \in R]| \leq \epsilon_1$ for $\epsilon_1 := 16c\epsilon$.
- By Claim 27 and (8), we have $\Pr_{q\sim\mathcal{Q}_x}[q \notin V_{\mathrm{YES}} \cup V_{\mathrm{NO}}] \leq \Pr_{q\sim\mathcal{Q}_x}[\mathcal{Q}_x(q) \notin (1 \pm \epsilon)\alpha 2^{-|q|}] \leq \epsilon_2$ for $\epsilon_2 := 100\epsilon$.

Therefore, from the correctness of the generalized Feigenbaum–Fortnow protocol (Theorem 18), our simulation protocol satisfies the completeness and soundness with probability at least $1 - 0.02 - (\frac{1}{8} + 2m(n)\epsilon_1 + 3m(n)\epsilon_2 + 3\delta) \geq \frac{3}{4}$, where the last inequality holds for some large constant c_0. □

This completes the proof of Theorem 17.

7 Non-Black-Box Selector for GapMCSP

In this section, we present an application of our proof techniques. For any oracle A and a constant $\epsilon > 0$, denote by $\mathrm{Gap}_\epsilon\mathrm{MCSP}^A$ the $2^{(1-\epsilon)n}$-factor approximation version of the Minimum A-Oracle Circuit Size Problem; that is, the problem is approximating the minimum A-oracle circuit of a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$ within a factor of $2^{(1-\epsilon)n}$, given the truth table of f. We construct a non-black-box selector for $\mathrm{Gap}_\epsilon\mathrm{MCSP}^A$ for any oracle A.

Theorem 31. *Let A, R be any oracles such that $R \in \mathsf{P}^A/\mathsf{poly}$.*[12] *Let $\alpha, c > 0$ be any constants. Then, there exist a randomized polynomial-time algorithm S and a constant $\beta > 0$ such that, for any $n \in \mathbb{N}$ and any R-oracle circuits C_0, C_1 of size n^c such that one of them computes $\mathrm{Gap}_\alpha\mathrm{MCSP}^A$ on every input of length at most n, for any instance $x \in \{0,1\}^n$ of $\mathrm{Gap}_\beta\mathrm{MCSP}^A$, $S^R(x, C_0, C_1)$ decides x correctly with high probability.*

We call S a "non-black-box" selector because the algorithm makes use of the property that two candidate algorithms solving $\mathrm{Gap}_\alpha\mathrm{MCSP}^A$ are modeled as polynomial-size circuits instead of oracles.

The non-black-box selector S uses the property that $\mathrm{GapMCSP}^A$ is reducible to any *polynomial-size* oracle avoiding the A-oracle circuit interpreter $G^{\mathsf{int},A}$, as captured in the following lemma.

Lemma 32 ($\mathrm{GapMCSP}^A$ Reduces to Natural Properties; [CIKK16, Hir18])**.** *Let A, R be any oracles. Let $\alpha > 0$ be any constant. Let $D = \{D_m\}_{m\in\mathbb{N}}$ be a family of R-oracle polynomial-size circuits D_m^R on 2^m inputs that $\frac{7}{8}$-avoids $G^{\mathsf{int},A}\colon \{0,1\}^{\widetilde{O}(2^{\alpha m})} \to \{0,1\}^{2^m}$. Then, there exist a constant $\beta > 0$ and a polynomial-time computable function G that takes a size parameter $s \in \mathbb{N}$, a truth table of a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$ and a string z and returns a string $G_s^f(z)$ of length $m = m(n,s) \le 2^n$ such that:*

1. *if $\mathsf{size}^A(f) \le s$, then $D(G_s^f(z)) = 0$ for every z, and*
2. *if $\mathsf{size}^R(f) \ge 2^{(1-\beta)n}s$, then $\Pr_z[D(G_s^f(z)) = 1] \ge \frac{3}{4}$.*

Proof. Let $f\colon \{0,1\}^n \to \{0,1\}$ be a Boolean function. We define G as the Nisan-Wigderson generator [NW94] instantiated with a hardness-amplified version of f as an underlying function. Specifically, let k be a parameter chosen later. We define a hardness-amplified version of f as $f^{\oplus k}(x^1, \cdots, x^k) = f(x^1) \oplus \cdots \oplus f(x^k)$ for $(x^1, \cdots, x^k) \in \{0,1\}^{nk}$.

We proceed to the definition of the Nisan-Wigderson generator NW: Let p be an arbitrary prime such that $kn \le p \le 2kn$ and $m \in \mathbb{N}$ be a parameter chosen later. For each string $q \in \{0,1\}^m$, we associate a polynomial q' of degree m over the field $\mathbb{F}_p$ defined as $q'(a) = \sum_{i=1}^m q_i a^i$ for any $a \in \mathbb{F}_p$. Take an arbitrary subset $D \subseteq \mathbb{F}_p$ of size kn, and define $S_q := \{\, (a, q'(a)) \mid a \in D \,\} \subseteq (\mathbb{F}_p)^2$. For a string $z \in \{0,1\}^{p^2}$, denote by $z_{S_q} \in \{0,1\}^{kn}$ the string obtained by concatenating the ith bit of z for every $i \in S_q$, identifying $i \in (\mathbb{F}_p)^2$ with $i \in [p^2]$. The function $G_s^f(z)$ is defined as $\mathrm{NW}^{f^{\oplus k}}(z) := (f(z_{S_q}) \,|\, q \in \{0,1\}^m) \in \{0,1\}^{2^m}$, that is, the truth table of the Boolean function that maps $q \mapsto f(z_{S_q})$. (We will choose k and m depending on the size parameter s.)

Consider any f such that $\mathsf{size}^A(f) \le s$. From the construction, it is easy to see that

$$\mathsf{size}^A(\mathrm{NW}^{f^{\oplus k}}(z)) \le \mathsf{poly}(\mathsf{size}^A(f), m, k, n) \le \mathsf{poly}(s, m, k, n)$$

[12] R is included for the purpose of stating the theorem as general as possible. The reader may think of $R := \varnothing$ for simplicity.

for every $z \in \{0,1\}^{p^2}$. For a sufficiently large m, this can be bounded above by $2^{\alpha m}$. Thus we can take m as the smallest integer such that $\mathsf{poly}(s,m,k,n) \le 2^{\alpha m}$. Since D avoids $G^{\mathsf{int},A}$, we obtain the first condition of Lemma 32 (i.e., $D_m(\mathrm{NW}^{f^{\oplus k}}(z)) = 0$).

Next, we prove the contrapositive of the second condition. We assume $\Pr_z[D_m^R(\mathrm{NW}^{f^{\oplus k}}(z)) = 1] < \frac{3}{4}$. Since D_m^R is $\frac{7}{8}$-dense, we have $\Pr_{w\sim\{0,1\}^m}[D_m^R(w)=1] \ge \frac{7}{8}$. In particular, D_m^R distinguishes $\mathrm{NW}^{f^{\oplus k}}$ from the uniform distribution with advantage $\frac{7}{8} - \frac{3}{4} = \frac{1}{8}$. By the security proof of the Nisan-Wigderson generator [NW94], we obtain an oracle circuit C such that $\Pr_{\bar{x}\sim\{0,1\}^{nk}}[C^{D_m^R}(\bar{x}) = f^{\oplus k}(\bar{x})] \ge \frac{1}{2} + \Omega(\frac{1}{2^m})$ and the size of C is $\mathsf{poly}(2^m)$. Since the size of the circuit D_m is at most a polynomial in the input length 2^m, we can replace oracle gates of C with D_m and obtain an R-oracle circuit C_1 of size $\mathsf{poly}(2^m)$ such that C_1^R computes $f^{\oplus k}$ with probability $\frac{1}{2} + \Omega(\frac{1}{2^m})$. By Yao's XOR lemma (cf. [GNW11]), we obtain a circuit C_2 of size $\mathsf{poly}(2^m, 1/\delta)$ such that $\Pr_{x\sim\{0,1\}^n}[C_2^R(x) = f(x)] \ge 1-\delta$, where δ is an arbitrary parameter that satisfies $(1-\delta)^k \le \Omega(\frac{1}{2^m})$. We define $k := \Theta(m/\delta)$ so that this inequality is satisfied, and we take $\delta := 2^{-\beta n}s/2n$ for some sufficiently small constant $\beta > 0$. Finally, we correct the error of C_2 in a trivial way: Let φ be a DNF formula of size at most $\delta n 2^n$ such that $\varphi(x) = 1$ if and only if $C_2^R(x) \ne f(x)$. Then a circuit $C_3^R := C_2^R \oplus \varphi$ is a circuit of size $\mathsf{poly}(2^m, 1/\delta) + \delta n 2^n$ that computes f exactly. By the definition of m, we have $2^m = \Theta(\mathsf{poly}(s,n,1/\delta))$, and thus $\mathsf{poly}(2^m, 1/\delta) \le \mathsf{poly}(s,n,1/\delta) \le (sn/\delta)^c$ for some constant c. Therefore, $\mathsf{size}^R(f) \le 2^{c\beta n}(2n)^{2c} + 2^{(1-\beta)n-1}s < 2^{(1-\beta)n}s$ for $\beta < 1/(c+1)$ and a sufficiently large n. □

Now we construct a non-black-box selector S for GapMCSP.

Proof of Theorem 31. The idea is exactly the same with Theorem 15. Namely, given two oracles R_0, R_1 such that one of them is guaranteed to be a dense subset of random strings, $R_0 \cap R_1$ is also a dense subset of random strings. We apply this idea to the case of GapMCSP. In our case, by a "random string" we mean a truth table that cannot be compressed into a small circuit.

Let $f\colon \{0,1\}^n \to \{0,1\}$ and $s \in \mathbb{N}$ be an instance of $\mathrm{Gap}_\beta\mathrm{MCSP}^A$, and $m = m(n,s)$ be as in Lemma 32. Let C_0, C_1 be R-oracle circuits one of which computes $\mathrm{Gap}_\alpha\mathrm{MCSP}^A$ for every instance of length 2^m ($\le 2^n$). For each $i \in \{0,1\}$, we fix the size parameter of the inputs to C_i to $2^{\alpha n/2}$; that is, define $C'^R_i(g) := C_i^R(g, 2^{\alpha n/2})$. When the ith circuit is correct, C'^R_i accepts every $g \in \{0,1\}^{2^m}$ such that $\mathsf{size}^A(g) \le 2^{\alpha n/2}$, and rejects every g such that $\mathsf{size}^A(g) > 2^{(1-\alpha)n+\alpha n/2} = 2^{(1-\alpha/2)n}$. In particular, for a random $g \sim \{0,1\}^{2^m}$, we have $\Pr[C'^R_i(g) = 1] \le o(1)$ by a simple counting argument.

Here is the algorithm S for solving $\mathrm{Gap}_\beta\mathrm{MCSP}^A$: For each $i \in \{0,1\}$, we verify that C_i has a small number of YES instances by sampling. Specifically, we pick $g \sim \{0,1\}^{2^m}$, verify that $C'^R_i(g) = 0$, and repeat this test $O(1)$ times. If one of these tests fails, it means that C'_i is not likely to compute $\mathrm{Gap}_\alpha\mathrm{MCSP}^A$ correctly; thus we redefine $C'_i := 0$. In particular, if $\Pr_{g\sim\{0,1\}^{2^m}}[C'^R_i(g) = 1] \ge \frac{1}{16}$, then with high probability C'_i is redefined. We then define a circuit D so that $D^R(g) := \neg(C'^R_0(g) \vee C'^R_1(g))$ for $g \in \{0,1\}^{2^m}$. By the test above, with high probability, D^R is dense, that is, $\Pr_g[D^R(g) = 1] \ge \frac{7}{8}$. Now let G be the function of Lemma 32. Pick a random z and accept if and only if $D(G_s^f(z)) = 0$.

We claim the correctness of the algorithm S described above. We verify that the hypothesis of Lemma 32 for $R = A$ is satisfied with high probability. Since $R \in \mathsf{P}^A/\mathsf{poly}$, the R-oracle polynomial-size circuit D can be simulated by an A-oracle polynomial-size circuit. Moreover, D^R

avoids $\mathrm{Im}(G^{\mathsf{int},A})$ since every $g \in \{0,1\}^{2^{\alpha n/2}}$ is rejected, and D^R is $\frac{7}{8}$-dense with high probability. Applying Lemma 32, the algorithm accepts every instance (f,s) such that $\mathsf{size}^A(f) \leq s$ and rejects every instance (f,s) such that $\mathsf{size}^A(f) \geq 2^{(1-\beta)n}s$, with high probability. □

The existence of a selector for a language L has several consequences. For example, if L is computable by a slightly-nonuniform algorithm, then by trying all the advice strings and selecting the correct computation by using the selector, one can get rid of the nonuniformity [Hir15]. It can also be shown that L is low for $\mathsf{S}_2^{\mathrm{p}}$ if $L \in \mathsf{P}/\mathsf{poly}$, which generalizes a result of [CCHO05] (from any downward self-reducible language to any language with a selector; a detail can be found in [Hir20c]). As a consequence of the non-black-box selector for GapMCSP, it is possible to prove the lowness of GapMCSP for $\mathsf{S}_2^{\mathrm{p}}$:

Theorem 33 (Lowness for $\mathsf{S}_2^{\mathrm{p}}$)**.** *Let A, R be any oracles such that $R \in \mathsf{P}^A/\mathsf{poly}$. Let $\alpha > 0$ be any constant. If $\mathrm{Gap}_\alpha\mathrm{MCSP}^A \in \mathsf{P}^R/\mathsf{poly}$, then $\mathsf{S}_2^{\mathrm{Gap}_\beta\mathrm{MCSP}^A} \subseteq \mathsf{S}_2^R$ for some constant $\beta > 0$.*

Proof Sketch. Let S be the non-black-box selector of Theorem 31. Let c be a constant such that there exists a R-oracle circuit C of size n^c such that C^R computes $\mathrm{Gap}_\alpha\mathrm{MCSP}^A$. Let $L \in \mathsf{S}_2^{\mathrm{Gap}_\beta\mathrm{MCSP}^A}$, and V be an S_2-type verifier for L. We present an $\mathsf{S}_2 \cdot \mathsf{BP} \cdot \mathsf{P}^R$ algorithm W: For each $i \in \{0,1\}$, the ith competing prover sends a polynomial-size R-oracle circuit C_i of size n^c and a certificate for V. Using the two certificates, W simulates V, and each query q that V makes is answered by running $S^R(q, C_0, C_1)$. The correctness of W follows from Theorem 31. Therefore, we have $L \in \mathsf{S}_2 \cdot \mathsf{BP} \cdot \mathsf{P}^R = \mathsf{S}_2^R$. □

The non-black-box property prevents us from generalizing Theorem 33 for every R. We leave it as an interesting open question whether Theorem 33 holds for any oracles R and A.

A Tightness of Our Simulation Algorithms

In this appendix, we present the evidence that the upper bounds of Theorem 4 are nearly tight. In Subsection A.1, we argue that the $\mathsf{AM} \cap \mathsf{coAM}$ upper bound cannot be significantly improved. In Subsection A.2, we show that our $\mathsf{S}_2^{\mathrm{p}}$-type simulation algorithm is tight in a certain case.

A.1 Security Proofs of a Hitting Set Generator Based on SZK

In this section, we show that the security of a hitting set generator can be proved by using a uniform black-box reduction, based on the worst-case hardness assumption of SZK.

Theorem 34. *Let $\epsilon > 0$ be any constant, and R be any oracle $\frac{1}{2}$-avoiding $G^{\mathsf{int}} = \{G_n^{\mathsf{int}} : \{0,1\}^{n^\epsilon} \to \{0,1\}^n\}_{n\in\mathbb{N}}$. Then, $\mathsf{SZK} \subseteq \mathsf{BPP}^R$.*

Proof Sketch. Take any problem $L \in \mathsf{SZK}$. Allender and Das [AD17] showed that there exists a candidate *auxiliary-input one-way function* f_L such that the task of solving L reduces to the task of inverting f_L. (This fact was first shown by Ostrovsky [Ost91]). Here we say that a randomized algorithm A inverts an auxiliary-input one-way function f_L if, for every auxiliary-input x, $\Pr_{A,y}[A(x, f_L(x,y)) \in f_L^{-1}(x,\text{-})] \geq 1/\mathsf{poly}(|x| + |y|)$. This task can be done by a randomized polynomial-time algorithm with oracle access to R, by constructing a candidate pseudorandom

(function) generator G^{f_L} based on f_L using [HILL99, GGM86], and observing that R can distinguish G^{f_L} from the uniform distribution. (See [ABK+06, Theorem 45], [AH17] for details.) □

The reduction of Theorem 34 is adaptive because of the use of [HILL99]. We conjecture that $\mathsf{SZK} \subseteq \bigcap_R \mathsf{BPP}_{\|}^R$, which implies that the $\mathsf{AM} \cap \mathsf{coAM}$ upper bound of Theorem 4 cannot be significantly improved. While we were not able to obtain nonadaptive reductions in the general case, we show that the problem of factoring the product of two primes is nonadaptively reducible to R.

Theorem 35. *Let $\epsilon > 0$ be any constant, and R be any oracle $\frac{1}{2}$-avoiding $G^{\mathsf{int}} = \{G_n^{\mathsf{int}} : \{0,1\}^{n^\epsilon} \to \{0,1\}^n\}_{n\in\mathbb{N}}$. Then, factoring the product of two primes can be done in $\mathsf{ZPP}_{\|}^R$.*

Proof. We follow the proof of [ABK+06, Theorem 47] showing that Integer Factorization is in $\mathsf{ZPP}^{\mathrm{MCSP}}$. The idea is that, by using [HILL99, GGM86], one can invert any auxiliary-input one-way function f_N with oracle access to R (cf. [ABK+06, Theorem 45], [AH17]). For the purpose of integer factorization, we can use some *regular* one-way function f_N, and we observe that the security reduction of [HILL99] is nonadaptive in this case.

Let $N \in \mathbb{N}$ be an input. We may assume without loss of generality that N is odd; hence, $N = pq$ for some odd prime numbers p, q. We may also assume that $p \neq q$. Let $f_N \colon (\mathbb{Z}/N\mathbb{Z})^* \to (\mathbb{Z}/N\mathbb{Z})^*$ be Rabin's (auxiliary-input) one-way function [Rab79]: $f_N(x) = x^2 \mod N$, where we regard N as an auxiliary input. As observed in [Rab79], f_N is a 4-to-1 function. (That is, for any a with $\gcd(a, N) = 1$, the number of $x \in (\mathbb{Z}/N\mathbb{Z})^*$ such that $x^2 \equiv a \pmod N$ is 4. Indeed, the condition is equivalent to saying that $x^2 \equiv a \pmod p$ and $x^2 \equiv a \pmod q$, and each congruence has exactly 2 solutions; by the Chinese remainder theorem, we obtain exactly 4 solutions.) Rabin showed that the task of factoring N is efficiently reducible to the task of inverting f_N on average. Therefore, it remains to show that one can efficiently invert f_N with parallel queries to R.

From the regular candidate one-way function f_N, a candidate pseudorandom generator $G_N = \{G_{N,n} : \{0,1\}^{n/2} \to \{0,1\}^n\}_{n\in\mathbb{N}}$ can be constructed as in [HILL99, Theorem 5.5] so that the reduction of the security proof for G_N is nonadaptive. By using a construction of [GGM86], we extend the output length of G_N so that $G'_N = \{G'_{N,n} : \{0,1\}^n \to \{0,1\}^{n^{O(1/\epsilon)}}\}_{n\in\mathbb{N}}$ with the property that $G'_{N,n}(z) \in \mathrm{Im}(G^{\mathsf{int}}_{n^{O(1/\epsilon)}})$ for every seed z (that is, the function represented by $G'_{N,n}(z)$ as a truth table can be computed by a circuit of size $n^{O(1)}$). Thus the oracle R distinguishes G'_N from the uniform distribution, and hence we obtain a nonadaptive randomized algorithm factoring N. Finally, the randomized algorithm can be made zero-error by using the primality test of [AKS04]. □

As an immediate corollary, we obtain a new hardness result of MCSP under nonadaptive reductions.

Corollary 36. *Factoring the product of two primes can be done in $\mathsf{ZPP}_{\|}^{\mathrm{Gap}_\epsilon \mathrm{MCSP}}$ for every constant $\epsilon > 0$.*

A.2 On the Simulation by Using the Competing Prover System

Our $\mathsf{S}_2^{\mathrm{p}}$-type simulation algorithm is in fact completely tight in a certain setting. Regard a universal Turing machine U as a hitting set generator. We consider an exponential-time analogue

of Theorem 4 when the reduction can make only short queries. Specifically, for an oracle R, denote by $\mathsf{BPEXP}^{R[\mathsf{poly}]}$ the class of languages that can be computed by a two-sided-error randomized $2^{n^{O(1)}}$-time algorithm that can query $q \overset{?}{\in} R$ of length $\leq n^{O(1)}$, on inputs of length n. (We note that all the queries of polynomial length can be asked by an exponential-time reduction, and thus the adaptivity does not matter here.) The computational power of $\bigcap_R \mathsf{BPEXP}^{R[\mathsf{poly}]}$ where R is an arbitrary oracle that avoids a universal Turing machine is *exactly* equal to the exponential-time analogue of $\mathsf{S}_2^{\mathrm{p}}$.

Theorem 37. *Regard a universal Turing machine U as a family of functions $U = \{U_\ell \colon \{0,1\}^{\ell/2} \to \{0,1\}^\ell\}_{\ell \in \mathbb{N}}$. Then,*

$$\bigcap_{R \text{ avoids } U} \mathsf{EXP}_{\|}^{R} \quad = \quad \bigcap_{R \text{ avoids } U} \mathsf{BPEXP}^{R[\mathsf{poly}]} \quad = \quad \mathsf{S}_2^{\mathrm{exp}}.$$

Proof. It was shown in [Hir20c] that $\bigcap_{R \text{ avoids } U} \mathsf{EXP}_{\|}^{R} = \bigcap_{R \text{ avoids } U} \mathsf{EXP}_{\|}^{R[\mathsf{poly}]} = \mathsf{S}_2^{\mathrm{exp}}$. It remains to claim that

$$\bigcap_{R \text{ avoids } U} \mathsf{BPEXP}^{R[\mathsf{poly}]} \quad \subseteq \quad \mathsf{S}_2^{\mathrm{exp}}.$$

Let $L \in \bigcap_{R \text{ avoids } U} \mathsf{BPEXP}^{R[\mathsf{poly}]}$. We first note that, as in Lemma 14, the order of quantifiers can be swapped; indeed, the proof of Lemma 14 does not rely on any specific property of $\mathsf{BPP}_{\|}$ reductions; hence, the same proof works for other notions of reduction. Thus, there exists some randomized $t(n)$-time black-box reduction M from a language L to any γ-avoiding oracle for U such that the length of any query that M makes on input of length n is at most $\log t(n)$, for some $t(n) = 2^{n^{O(1)}}$. Let $L' := \{\, x01^{t(n)} \mid x \in L \,\}$ be a padded version of L. Applying Theorem 15 to L', we obtain $L' \in \mathsf{S}_2^{\mathrm{p}}$, from which it follows that $L \in \mathsf{S}_2^{\mathrm{exp}}$. □

References

[ABK+06] Eric Allender, Harry Buhrman, Michal Koucký, Dieter van Melkebeek, and Detlef Ronneburger. Power from Random Strings. *SIAM J. Comput.*, 35(6):1467–1493, 2006.

[ABX08] Benny Applebaum, Boaz Barak, and David Xiao. On Basing Lower-Bounds for Learning on Worst-Case Assumptions. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 211–220, 2008.

[AD17] Eric Allender and Bireswar Das. Zero knowledge and circuit minimization. *Inf. Comput.*, 256:2–8, 2017.

[Adl78] Leonard M. Adleman. Two Theorems on Random Polynomial Time. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 75–83, 1978.

[AGGM06] Adi Akavia, Oded Goldreich, Shafi Goldwasser, and Dana Moshkovitz. On basing one-way functions on NP-hardness. In *Proceedings of the Symposium on Theory of Computing (STOC)*, pages 701–710, 2006.

[AGGM10] Adi Akavia, Oded Goldreich, Shafi Goldwasser, and Dana Moshkovitz. Erratum for: on basing one-way functions on NP-hardness. In *Proceedings of the Symposium on Theory of Computing (STOC)*, pages 795–796, 2010.

[AH17] Eric Allender and Shuichi Hirahara. New Insights on the (Non-)Hardness of Circuit Minimization and Related Problems. In *Proceedings of the International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 54:1–54:14, 2017.

[Ajt96] Miklós Ajtai. Generating Hard Instances of Lattice Problems (Extended Abstract). In *Proceedings of the Symposium on the Theory of Computing (STOC)*, pages 99–108, 1996.

[AKS04] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. PRIMES is in P. *Annals of mathematics*, pages 781–793, 2004.

[All12] Eric Allender. Curiouser and Curiouser: The Link between Incompressibility and Complexity. In *Proceedings of the 8th Conference on Computability in Europe (CiE)*, pages 11–16, 2012.

[Bab85] László Babai. Trading Group Theory for Randomness. In *Proceedings of the Symposium on Theory of Computing (STOC)*, pages 421–429, 1985.

[BB15] Andrej Bogdanov and Christina Brzuska. On Basing Size-Verifiable One-Way Functions on NP-Hardness. In *Proceedings of the Theory of Cryptography Conference (TCC)*, pages 1–6, 2015.

[BT06a] Andrej Bogdanov and Luca Trevisan. Average-Case Complexity. *Foundations and Trends in Theoretical Computer Science*, 2(1), 2006.

[BT06b] Andrej Bogdanov and Luca Trevisan. On Worst-Case to Average-Case Reductions for NP Problems. *SIAM J. Comput.*, 36(4):1119–1159, 2006.

[Can96] Ran Canetti. More on BPP and the Polynomial-Time Hierarchy. *Inf. Process. Lett.*, 57(5):237–241, 1996.

[CCHO05] 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.

[CIKK16] Marco L. Carmosino, Russell Impagliazzo, Valentine Kabanets, and Antonina Kolokolova. Learning Algorithms from Natural Proofs. In *Proceedings of the Conference on Computational Complexity (CCC)*, pages 10:1–10:24, 2016.

[FF93] Joan Feigenbaum and Lance Fortnow. Random-Self-Reducibility of Complete Sets. *SIAM J. Comput.*, 22(5):994–1005, 1993.

[For89] Lance Fortnow. The Complexity of Perfect Zero-Knowledge. *Advances in Computing Research*, 5:327–343, 1989.

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986.

[GNW11] Oded Goldreich, Noam Nisan, and Avi Wigderson. On Yao's XOR-Lemma. In *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness*

and Computation - In Collaboration with Lidor Avigad, Mihir Bellare, Zvika Brakerski, Shafi Goldwasser, Shai Halevi, Tali Kaufman, Leonid Levin, Noam Nisan, Dana Ron, Madhu Sudan, Luca Trevisan, Salil Vadhan, Avi Wigderson, David Zuckerman, pages 273–301. 2011.

[GS86] Shafi Goldwasser and Michael Sipser. Private Coins versus Public Coins in Interactive Proof Systems. In *Proceedings of the Symposium on Theory of Computing (STOC)*, pages 59–68, 1986.

[GV08] Dan Gutfreund and Salil P. Vadhan. Limitations of Hardness vs. Randomness under Uniform Reductions. In *Proceedings of the Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, pages 469–482, 2008.

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A Pseudorandom Generator from any One-way Function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

[Hir15] Shuichi Hirahara. Identifying an Honest $\mathsf{EXP}^{\mathsf{NP}}$ Oracle Among Many. In *Proceedings of the Conference on Computational Complexity (CCC)*, pages 244–263, 2015.

[Hir18] Shuichi Hirahara. Non-black-box Worst-case to Average-case Reductions within NP. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 247–258, 2018.

[Hir20a] Shuichi Hirahara. Non-Disjoint Promise Problems from Meta-Computational View of Pseudorandom Generator Constructions. To appear in CCC'20, 2020.

[Hir20b] Shuichi Hirahara. Unexpected Hardness Results for Kolmogorov Complexity Under Uniform Reductions. *To appear in STOC'20*, 2020.

[Hir20c] Shuichi Hirahara. Unexpected Power of Random Strings. In *Proceedings of the Innovations in Theoretical Computer Science Conference (ITCS)*, pages 41:1–41:13, 2020.

[Hoe63] Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association*, 58(301):13–30, 1963.

[Hol05] Thomas Holenstein. Key agreement from weak bit agreement. In *Proceedings of the Symposium on Theory of Computing (STOC)*, pages 664–673, 2005.

[Hol06] Thomas Holenstein. Pseudorandom Generators from One-Way Functions: A Simple Construction for Any Hardness. In *Proceedings of the Theory of Cryptography Conference (TCC)*, pages 443–461, 2006.

[HRV13] Iftach Haitner, Omer Reingold, and Salil P. Vadhan. Efficiency Improvements in Constructing Pseudorandom Generators from One-Way Functions. *SIAM J. Comput.*, 42(3):1405–1430, 2013.

[HS17] Shuichi Hirahara and Rahul Santhanam. On the Average-Case Complexity of MCSP and Its Variants. In *Proceedings of the Computational Complexity Conference (CCC)*, pages 7:1–7:20, 2017.

[HW16] Shuichi Hirahara and Osamu Watanabe. Limits of Minimum Circuit Size Problem as Oracle. In *Proceedings of the Conference on Computational Complexity (CCC)*, pages 18:1–18:20, 2016.

[Imp11] Russell Impagliazzo. Relativized Separations of Worst-Case and Average-Case Complexities for NP. In *Proceedings of the Conference on Computational Complexity (CCC)*, pages 104–114, 2011.

[IW01] Russell Impagliazzo and Avi Wigderson. Randomness vs Time: Derandomization under a Uniform Assumption. *J. Comput. Syst. Sci.*, 63(4):672–688, 2001.

[KC00] Valentine Kabanets and Jin-yi Cai. Circuit minimization problem. In *Proceedings of the Symposium on Theory of Computing (STOC)*, pages 73–79, 2000.

[Lev86] Leonid A. Levin. Average Case Complete Problems. *SIAM J. Comput.*, 15(1):285–286, 1986.

[NW94] Noam Nisan and Avi Wigderson. Hardness vs Randomness. *J. Comput. Syst. Sci.*, 49(2):149–167, 1994.

[Ost91] Rafail Ostrovsky. One-Way Functions, Hard on Average Problems, and Statistical Zero-Knowledge Proofs. In *Proceedings of the Structure in Complexity Theory Conference*, pages 133–138, 1991.

[PSV06] Aduri Pavan, Rahul Santhanam, and N. V. Vinodchandran. Some Results on Average-Case Hardness Within the Polynomial Hierarchy. In *Proceedings of the Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 188–199, 2006.

[Rab79] Michael O. Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical report, Cambridge, MA, USA, 1979.

[RS98] Alexander Russell and Ravi Sundaram. Symmetric Alternation Captures BPP. *Computational Complexity*, 7(2):152–162, 1998.

[Rud97] Steven Rudich. Super-bits, Demi-bits, and NP/qpoly-natural Proofs. In *Proceedings of the Randomization and Approximation Techniques in Computer Science (RANDOM/APPROX)*, pages 85–93, 1997.

[San09] Rahul Santhanam. Circuit Lower Bounds for Merlin–Arthur Classes. *SIAM J. Comput.*, 39(3):1038–1061, 2009.

[STV01] Madhu Sudan, Luca Trevisan, and Salil P. Vadhan. Pseudorandom Generators without the XOR Lemma. *J. Comput. Syst. Sci.*, 62(2):236–266, 2001.

[TV07] Luca Trevisan and Salil P. Vadhan. Pseudorandomness and Average-Case Complexity Via Uniform Reductions. *Computational Complexity*, 16(4):331–364, 2007.

[Vio05] Emanuele Viola. The complexity of constructing pseudorandom generators from hard functions. *Computational Complexity*, 13(3-4):147–188, 2005.

[Wat12] Thomas Watson. Relativized Worlds without Worst-Case to Average-Case Reductions for NP. *TOCT*, 4(3):8:1–8:30, 2012.

[Yao82] Andrew Chi-Chih Yao. Theory and Applications of Trapdoor Functions (Extended Abstract). In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 80–91, 1982.

[Yap83] Chee-Keng Yap. Some Consequences of Non-Uniform Conditions on Uniform Classes. *Theor. Comput. Sci.*, 26:287–300, 1983.

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il