Electronic Colloquium on Computational Complexity, Report No. 50 (2020)

Unexpected Hardness Results for Kolmogorov Complexity Under Uniform Reductions

Shuichi Hirahara
National Institute of Informatics
s_hirahara@nii.ac.jp

April 18, 2020

Abstract

Hardness of computing the Kolmogorov complexity of a given string is closely tied to a security proof of hitting set generators, and thus understanding hardness of Kolmogorov complexity is one of the central questions in complexity theory. In this paper, we develop new proof techniques to show hardness of computing Kolmogorov complexity under *surprisingly efficient reductions*, which were previously conjectured to be impossible. It is known that the set R_{K} of Kolmogorov-random strings is PSPACE-hard under polynomial-time Turing reductions, i.e., $\mathsf{PSPACE} \subseteq \mathsf{P}^{R_{\mathrm{K}}}$, and that $\mathsf{NEXP} \subseteq \mathsf{NP}^{R_{\mathrm{K}}}$, which was conjectured to be tight by Allender [All12]. We prove that $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathsf{P}^{R_{\mathrm{K}}}$, which simultaneously improves these hardness results and refutes the conjecture of Allender under the plausible assumption that $\mathsf{EXP}^{\mathsf{NP}} \neq \mathsf{NEXP}$. At the core of our results is a new security proof of a pseudorandom generator via a black-box uniform reduction, which overcomes an impossibility result of Gutfreund and Vadhan [GV08].

Our proof techniques have further consequences, including:

1. Applying our proof techniques to the case of resource-bounded Kolmogorov complexity, we obtain NP-hardness of the problem $\mathrm{MINcKT}^{\mathsf{SAT}}$ of computing conditional polynomial-time-bounded SAT-oracle Kolmogorov complexity under polynomial-time deterministic reductions. In contrast, the Minimum SAT-Oracle Circuit Size Problem cannot be NP-hard under polynomial-time deterministic reductions without resolving $\mathsf{EXP} \neq \mathsf{ZPP}$. Our hardness result is the first result that overcomes the non-NP-hardness results of MCSP. We also prove DistNP-hardness of $\mathrm{MINKT}^{\mathsf{SAT}}$, which is a partial converse of the approach of Hirahara [Hir18] for proving the equivalence between worst-case and average-case complexity of NP.
2. We prove $\mathsf{S}_2^{\mathrm{p}}$-hardness of Kolmogorov complexity under quasi-polynomial-time *nonadaptive* reductions. This is the first result that overcomes a $\mathsf{P/poly}$ barrier result of Allender, Buhrman, Friedman, and Loff [ABFL14].

We also establish a firm link between non-trivial satisfiability algorithms and the immunity of random strings, and we obtain the following unconditional lower bounds.

1. It has been a long-standing open question whether the set of subexponential-time-bounded Kolmogorov-random strings is decidable in P. We resolve this open question, by showing that the set of super-polynomial-time-bounded Kolmogorov-random strings is P-*immune*, which is a much stronger lower bound than an average-case lower bound.
2. The set of Levin's Kolmogorov-random strings is (P-uniform ACC^0)-immune.

ISSN 1433-8092

Contents

1 Introduction **1**

2 Kolmogorov-Randomness versus Pseudorandomness **2**
2.1 Conjectures on Limits of Hardness of Kolmogorov-Random Strings 3

3 Overview of Our Results **4**
3.1 NP-Hardness of SAT-Oracle MINcKT 5
3.2 Hardness of Kolmogorov Complexity Under Nonadaptive Reductions 7
3.3 Non-Trivial Satisfiability Algorithms and Immunity of Random Strings 8
3.4 Proof Techniques 9
3.5 Summary and Concluding Remarks 11

4 Preliminaries **12**

5 Randomized Enumeration Lemma for Functions **13**

6 Hardness Under Randomized Reductions **16**

7 Deterministic Enumeration Lemma for Strings **19**

8 Hardness Under Deterministic Reductions **21**
8.1 Hardness of MINcKT with SAT Oracle 21
8.2 Symmetry of Information and Hardness of Kolmogorov Complexity 24
8.3 Average-Case NP-Hardness of SAT-Oracle MINKT 25

9 Satisfiability Algorithms and Immunity of Random Strings **27**

1 Introduction

The holy grail of complexity theory is to separate complexity classes such as P and NP. Owing to the Cook–Levin theorem [Coo71, Lev73] which shows the NP-completeness of SAT, the P versus NP question reduces to the question of proving a lower bound for an arbitrary NP-complete problem. As reported in [BM97, BT00], Kolmogorov, however, suggested that

> NP-complete problems are probably too structured to be good candidates for separating P from NP. One should rather focus on the intermediate less structured sets that somehow are complex enough to prove separations.

As a specific candidate of less structured sets, he proposed to consider the set of *resource-bounded Kolmogorov-random strings*.

The *Kolmogorov complexity* $\mathrm{K}(x)$ of a finite string $x \in \{0,1\}^*$ is defined as the length of the shortest program that prints x. More formally, fix a universal Turing machine U and define $\mathrm{K}_U(x)$ as the length of the shortest program d such that U outputs x on input d; the subscript U is often omitted. Given a time bound $t\colon \mathbb{N} \to \mathbb{N}$, the *$t$-time-bounded Kolmogorov complexity* $\mathrm{K}^t(x)$ of a string x is defined as the length of the shortest program that prints x in $t(|x|)$ steps. The notion of Kolmogorov complexity induces the notion of randomness of a finite string. We say that a string x is *random* (with respect to K^t) if $\mathrm{K}^t(x) \geq |x| - 1$; we denote by R_{K^t} the set of all random strings.

The approach of Kolmogorov can be stated as proving that $R_{\mathrm{K}^t} \notin \mathsf{P}$ for some polynomial time bound $t(n) = n^{O(1)}$. Note here that R_{K^t} is a problem in coNP; thus, $R_{\mathrm{K}^t} \notin \mathsf{P}$ implies that $\mathsf{P} \neq \mathsf{NP}$.

Since the introduction of resource-bounded Kolmogorov-random strings ([Sip83, Har83, Ko86]), the set of resource-bounded Kolmogorov-random strings has been well investigated. However, the approach of Kolmogorov has not been fulfilled even for sub-exponential-time-bounded Kolmogorov complexity. It is easy to see that $R_{\mathrm{K}^t} \notin \mathsf{P}$ for any time bound $t(n) = 2^{n^c}$ for any constant $c > 1$. (Indeed, if $R_{\mathrm{K}^t} \in \mathsf{P}$, then one could print a random string of length n by an exhaustive search in time $2^{n+O(\log n)}$, which is a contradiction.) For any constant $0 < \epsilon \leq 1$, it has been a long-standing open question[1] to prove $R_{\mathrm{K}^t} \notin \mathsf{P}$ for a time bound $t(n) := 2^{n^\epsilon}$.

One of our results is to resolve this open question.

Theorem 1.1. *$R_{\mathrm{K}^t} \notin \mathsf{P}$ for any super-polynomial time bound $t(n) = n^{\omega(1)}$.*

A natural approach for showing Theorem 1.1 is to present an efficient reduction from any problem $\mathsf{DTIME}(n^{\omega(1)})$ to R_{K^t}. For example, it is easy to see that any EXP-complete problem under polynomial-time Turing reductions is not computable in P using the time hierarchy theorem [HS65]. Previously, it was known that R_{K^t} is EXP-complete under non-uniform polynomial-time reductions or NP-Turing reductions for $t(n) = 2^{n^\epsilon}$ with a constant $\epsilon > 0$ [ABK+06b]; however, these reductions are not efficient enough to prove the unconditional lower bound of Theorem 1.1 (because $\mathsf{EXP} \not\subseteq \mathsf{P/poly}$ and $\mathsf{EXP} \not\subseteq \mathsf{NP}$ are open).

We herein develop new proof techniques for showing hardness of computing Kolmogorov complexity under *surprisingly efficient reductions*, which were previously conjectured to be impossible. In Section 2, we present the importance of understanding hardness of Kolmogorov complexity while reviewing the literature. We state our results in Section 3 in more detail.

[1] According to Eric Allender and Osamu Watanabe (personal communication).

2 Kolmogorov-Randomness versus Pseudorandomness

A line of work uncovered a close relationship between Kolmogorov-randomness and pseudorandomness, or more specifically, the equivalence between hardness of deciding the set of Kolmogorov-random strings and a security proof of hitting set generators. A family of functions $G = \{G_n : \{0,1\}^{s(n)} \to \{0,1\}^n\}_{n\in\mathbb{N}}$ is called a *hitting set generator* secure against a circuit class $\mathfrak{C}$ if, for every n-input circuit $C \in \mathfrak{C}$ such that C accepts at least a half of n-bit inputs, there exists a seed $z \in \{0,1\}^{s(n)}$ such that C accepts $G(z)$. We say that an algorithm C is a *distinguisher* for a hitting set generator G if C violates this condition, i.e., C rejects every string in the range of G whereas C accepts at least a half of strings. The existence of an efficient hitting set generator enables us to derandomize a one-sided-error randomized $\mathfrak{C}$-algorithm by simply trying all seeds $z \in \{0,1\}^{s(n)}$ and using $G(z)$ as a source of randomness.

The following two directions exhibit the close connection between Kolmogorov-randomness and pseudorandomness.

1. (Kolmogorov-randomness is easy $\implies$ no pseudorandomness)

   The set R_{K} of Kolmogorov-random strings is a distinguisher for a hitting set generator. Indeed, for any computable hitting set generator $G = \{G_n\}_{n\in\mathbb{N}}$, the Kolmogorov complexity of $G_n(z)$ is small: $\mathrm{K}(G_n(z)) \leq s(n) + O(\log n)$ for any $z \in \{0,1\}^{s(n)}$; thus $G_n(z) \notin R_{\mathrm{K}}$. On the other hand, by a simple counting argument, most strings are random: $w \in R_{\mathrm{K}}$ holds with high probability over the choice of a uniformly at random w.

   Allender, Buhrman, Koucký, van Melkebeek, and Ronneburger [ABK+06b] exploited this fact and presented a number of hardness results of Kolmogorov complexity.

2. (no pseudorandomness $\implies$ Kolmogorov-randomness is easy)

   Conversely, any efficient distinguisher for a hitting set generator gives rise to an efficient algorithm for "approximating" the set of Kolmogorov-random strings. Hirahara [Hir18] proved this direction for time-bounded Kolmogorov complexity via a non-black-box reduction.

According to Item 1, any computable hitting set generator construction whose security is based on hardness of some function f gives rise to an efficient reduction from f to R_{K}. Allender et al. [ABK+06b] used a pseudorandom generator[2] constructed by Babai, Fortnow, Nisan, and Wigderson [BFNW93] to show that the halting problem is reducible to R_{K} via polynomial-size circuit reductions. Using the pseudorandom generator constructed by Trevisan and Vadhan [TV07] based on a uniform hardness assumption on a PSPACE-complete problem, they also showed that $\mathsf{PSPACE} \subseteq \mathsf{P}^{R_{\mathrm{K}}}$, which improves an earlier result of Buhrman and Torenvliet [BT00]. Similarly, Allender, Buhrman, and Koucký [ABK06a] showed that $\mathsf{NEXP} \subseteq \mathsf{NP}^{R_{\mathrm{K}}}$.

According to Item 2, the approach taken by Allender et al. [ABK+06b] for showing hardness of R_{K} is likely to be a universal approach — at least in a resource-bounded and non-black-box reduction setting. In particular, any hardness result of approximating time-bounded Kolmogorov complexity can be converted to a security proof of some hitting set generator.

To summarize, understanding hardness of Kolmogorov complexity is central to the study of pseudorandomness because it clarifies the types of problems whose hardness can give rise to a hitting set generator.

[2] A *pseudorandom generator* is a stronger notion than a hitting set generator that can derandomize a two-sided-error randomized algorithm.

2.1 Conjectures on Limits of Hardness of Kolmogorov-Random Strings

The PSPACE-hardness and NEXP-hardness results mentioned above were conjectured to be "tight" in a certain formal sense. Allender [All12] conjectured that the NEXP-hardness result of R_{K} under NP-Turing reductions is tight; Allender, Buhrman, and Koucký [ABK06a] speculated that PSPACE could be characterized by an efficient reduction to R_{K_U} by "factoring out" a particular choice of *universal Turing machine* U; here, a machine U is said to be *universal* if, for any machine M, there exists a constant c such that $\mathrm{K}_U(x) \leq \mathrm{K}_M(x) + c$ for any $x \in \{0,1\}^*$. The hardness results mentioned before hold for an *arbitrary* universal Turing machine. That is, for any universal Turing machine U, the following hold.

- $\mathsf{PSPACE} \subseteq \mathsf{P}^{R_{\mathrm{K}_U}}$ [ABK+06b].
- $\mathsf{NEXP} \subseteq \mathsf{NP}^{R_{\mathrm{K}_U}}$ [ABK06a].
- $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathsf{P}^{\mathsf{NP}^{R_{\mathrm{K}_U}}}$ and $\mathsf{EXPH} \subseteq \mathsf{PH}^{R_{\mathrm{K}_U}}$ [Hir20].

A typical pattern in these previous results is that the set of random strings provides (at most) exponential-time speed up. The conjectures of [ABK06a, All12] can be formally stated as follows.[3]

Hypothesis 2.1 (Allender, Buhrman, and Koucký [ABK06a])**.**

$$\mathsf{PSPACE} = \bigcap_U \mathsf{P}^{R_{\mathrm{K}_U}}.$$

Conjecture 2.2 (Allender [All12])**.**

$$\mathsf{NEXP} = \bigcap_U \mathsf{NP}^{R_{\mathrm{K}_U}}.$$

Here, the intersections are taken over all universal Turing machines U.

Allender, Friedman, and Gasarch [AFG13] (together with the result of [CDE+14]) showed that $\bigcap_U \mathsf{NP}^{R_{\mathrm{K}_U}} \subseteq \mathsf{EXPSPACE}$, where the intersection is taken over all prefix-free universal Turing machines.[4] That is,

$$\mathsf{PSPACE} \subseteq \bigcap_U \mathsf{P}^{R_{\mathrm{K}_U}} \subseteq \mathsf{EXPSPACE}.$$

Conjecture 2.2 states that the EXPSPACE upper bound is probably too loose, and that the class $\bigcap_U \mathsf{P}^{R_{\mathrm{K}_U}}$ is much closer to the lower bound. The reader is referred to a recent survey of Allender [All17] for more background.

While the original work of [ABK06a] provided little evidence supporting Hypothesis 2.1, their hypothesis is in fact quite plausible from the perspective of derandomization under uniform hardness assumptions. Subsequent to [ABK06a], Gutfreund and Vadhan [GV08] showed that the pseudo-random generator construction of Trevisan and Vadhan [TV07] is not likely to be improved beyond PSPACE. Specifically, they showed that there exists no hitting set generator whose security can be based on any problem outside PSPACE/poly via a black-box mildly-adaptive reduction.

[3] While the authors of [ABK06a] speculated that Hypothesis 2.1 might be true, they stopped short of calling it a "conjecture."

[4] For the sake of simplicity, we ignore a potential difference between prefix-free Kolmogorov complexity and plain Kolmogorov complexity because Hirahara and Kawamura [HK18] proved that the same EXPSPACE upper bound can be obtained for *plain* universal Turing machines by imposing a very mild restriction on a reduction.

Theorem 2.3 (Limits of a Black-Box Security Proof; Gutfreund and Vadhan [GV08])**.** *Let $L \notin \mathsf{PSPACE/poly}$ be an arbitrary problem. Then, there exists no hitting set generator $G = \{G : \{0,1\}^{s(n)} \to \{0,1\}^n\}_{n\in\mathbb{N}}$ with $s(n) < n/4$ such that there exists a randomized polynomial-time mildly-adaptive reduction from L to R, where R is an arbitrary distinguisher for G.*

This results can be seen as evidence supporting Hypothesis 2.1: There is a $\mathsf{PSPACE/poly}$ upper bound on the class of problems that are mildly-adaptive-reducible to an arbitrary distinguisher R (instead of R_{K_U}) for a hitting set generator, which is a subclass of $\bigcap_U \mathsf{P}^{R_{\mathrm{K}_U}}$.[5]

3 Overview of Our Results

In this work, we develop new proof techniques for showing the security of a hitting set generator under surprisingly efficient reductions, which overcome the impossibility result of Gutfreund and Vadhan.[6] We prove

Theorem 3.1 (Unexpected Power of Polynomial-Time Reductions to R_{K})**.**

$$\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathsf{P}^{R_{\mathrm{K}}}.$$

Theorem 3.1 shows $\mathsf{EXP}^{\mathsf{NP}}$-hardness of R_{K} under polynomial-time Turing reductions, which were conjectured to be impossible in light of Conjecture 2.2. More specifically, Theorem 3.1 refutes both Hypothesis 2.1 and Conjecture 2.2 simultaneously under the plausible assumption that $\mathsf{EXP}^{\mathsf{NP}} \neq \mathsf{NEXP}$. It is also worth mentioning that all of the previous results [ABK+06b, ABK06a, Hir20] on hardness of R_{K} under uniform reductions provided at most exponential-time speed up by using efficient PCP-type proof systems; Theorem 3.1 is the first result showing that R_{K} provides more than exponential-time speed up.

How do we overcome the impossibility result of Gutfreund and Vadhan? The point is that the impossibility result of Theorem 2.3 is known to hold only for $s(n) < n/4$, that is, a hitting set generator that stretches its seed by a factor of 4. At the core of Theorem 3.1 is the following new construction of a pseudorandom generator that stretches its seed by a small amount; its security is based on a uniform hardness assumption on $\mathsf{EXP}^{\mathsf{NP}}$ and can be proved by a black-box reduction.

Theorem 3.2. *There exists an $\mathsf{EXP}^{\mathsf{NP}}$-computable pseudorandom generator*

$$G = \{G_n : \{0,1\}^{n-O(\log n)} \to \{0,1\}^n\}_{n\in\mathbb{N}}$$

such that $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathsf{BPP}^D$ for any distinguisher D for G.

It is worth mentioning that Theorem 3.2 is nearly tight. Indeed, in our previous work [Hir20], we showed an almost matching upper bound $\bigcap_D \mathsf{BPP}^D \subseteq \mathsf{S}_2^{\mathrm{exp}}$, where the intersection is taken over all distinguishers D for an arbitrary function G. Here, $\mathsf{S}_2^{\mathrm{exp}}$ is the exponential-time analogue of the class $\mathsf{S}_2^{\mathrm{p}}$ of problems that admit a two-competing-prover system, and it is known that $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathsf{S}_2^{\mathrm{exp}} \subseteq \mathsf{ZPEXP}^{\mathsf{NP}}$ ([RS98, Cai07]). In the same paper [Hir20], we proved a much weaker result

[5] This upper bound holds even if the reduction is assumed to be randomized because $\mathsf{P}^{R_{\mathrm{K}_U}} = \mathsf{BPP}^{R_{\mathrm{K}_U}}$ [ABK+06b].

[6] Impagliazzo and Wigderson [IW01] constructed a pseudorandom generator based on a uniform hardness assumption of EXP. As noted in [GV08], their security proof is a non-black-box reduction, and thus the result of [IW01] does not refute Conjecture 2.2. In contrast, our focus here is a black-box security proof.

showing that $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathsf{P}^{\mathsf{NP}^{R_{\mathrm{K}}}}$, which is not sufficient for refuting Conjecture 2.2; Theorem 3.1 significantly improves the reducibility notion from P^{NP}-Turing reductions to P-Turing reductions.

Theorem 3.2 is not subject to the impossibility result of Theorem 2.3 because of the following two reasons. First, the seed length of our pseudorandom generator is close to the output length. Second, the security reduction of Theorem 3.2 is not necessarily mildly adaptive; however, the latter reason does not seem to be essential. Indeed, even in the case of *nonadaptive* reductions, we will show that NEXP and coNEXP are reducible to R_{K}. Hirahara and Watanabe [HW19] improved the impossibility result of Theorem 2.3 to $\mathsf{NP/poly} \cap \mathsf{coNP/poly}$ for nonadaptive reductions; therefore, our proof techniques overcome the impossibility result because our pseudorandom generator extends its seed by a small amount (unless $\mathsf{NEXP} \subseteq \mathsf{NP/poly} \cap \mathsf{coNP/poly}$).

3.1 NP-Hardness of SAT-Oracle MINcKT

Our proof techniques behind Theorem 3.1 are applicable to a wide range of questions about hardness of Kolmogorov complexity, including *resource-bounded* Kolmogorov complexity.

Understanding hardness of time-bounded Kolmogorov complexity is closely connected to one of the central questions of complexity theory — an equivalence between average-case and worst-case complexity of NP. Hirahara [Hir18] showed that NP-completeness of GapMINKT implies an equivalence between average-case and worst-case complexity of NP. Here, GapMINKT is an approximation version of the problem of deciding whether $\mathrm{K}^t(x) \leq s$ given $(x, 1^t, 1^s)$ as input. The proof technique of [Hir18] is not subject to the well-known black-box reduction barrier of Bogdanov and Trevisan [BT06], and thus, proving NP-completeness of GapMINKT is a reasonable approach for establishing an equivalence between average-case and worst-case complexity of NP.

We herein establish NP-hardness of a problem closely related to MINKT, thereby making a progress towards better understanding of hardness of MINKT.

Theorem 3.3 (NP-hardness of a conditional version of $\mathrm{MINKT}^{\mathsf{SAT}}$)**.**

$$\mathrm{MINcKT}^{\mathsf{SAT}} \textit{ is } \mathsf{P}^{\mathsf{NP}}\textit{-hard under polynomial-time nonadaptive reductions.}$$

Here, $\mathrm{MINcKT}^{\mathsf{SAT}}$ is a problem of deciding whether $\mathrm{K}^{t,\mathsf{SAT}}(x|y) \leq s$ given $(x, y, 1^t, 1^s)$ as input, where $\mathrm{K}^{t,\mathsf{SAT}}(x|y)$ is the minimum size of a program that outputs x given y as input and oracle access to SAT; we assume that t is "sufficiently large," as otherwise K^t might largely depend on a particular choice of universal machines.[7] (We defer a precise definition of $\mathrm{MINcKT}^{\mathsf{SAT}}$ to Section 8.1.) It is easy to see that $\mathrm{MINcKT}^{\mathsf{SAT}}$ is a problem in $\Sigma_2^{\mathrm{p}} = \mathsf{NP}^{\mathsf{NP}}$.

Now we review several results related to Theorem 3.3, and then explain its importance. A closely related question is NP-hardness of the Minimum Circuit Size Problem (MCSP), which is the problem of asking, given the truth table of a Boolean function f and a size parameter s as input, whether there exists a circuit of size at most s that computes f. The minimum circuit size of a function represented by a string x can be regarded as a "sublinear-time-bounded Kolmogorov complexity" of x, in the sense that each bit of a string can be efficiently described (cf. KT-complexity [ABK+06b]). While NP-hardness of MCSP is widely open, there has been a line of work which proves NP-hardness of variants of MCSP, such as MCSP for DNF formulas [Mas79, AHM+08], MCSP for $\mathsf{DNF} \circ \mathsf{XOR}$ circuits [HOS18], and an oracle variant of MCSP [Ila20]. These results

[7] We mention that Vazirani and Vazirani [VV83] proved NP-hardness of a version of MINcKT for a specific choice of U and a time bound t fixed to be linear.

exploit the fact that an underlying computational model is "weak" in the sense that one can prove some lower bound (for $\mathsf{DNF} \circ \mathsf{XOR}$ circuits or a limited access to an oracle).

In contrast, our proofs exploit the fact that the underlying computational model is *strong* in the sense that a satisfying assignment is computable in $\mathsf{P}^{\mathsf{SAT}}$. Our proof techniques are more relevant to a result of Allender et al. [ABK+06b], which shows PSPACE-completeness of $\mathrm{MCSP}^{\mathsf{QBF}}$ (and $\mathrm{MINKT}^{\mathsf{QBF}}$) under ZPP-Turing reductions, where QBF is a PSPACE-complete problem. Impagliazzo, Kabanets, and Volkovich [IKV18] generalized it to A-hardness of MCSP^A for complete problems A of several complexity classes (such as PP, $\#\mathsf{P}$, and $\oplus\mathsf{P}$). It has been an open question to present a strong hardness result[8] of MCSP^A, MINKT^A or MINcKT^A for any oracle A within PH. Theorem 3.3 resolves this question for $\mathrm{MINcKT}^{\mathsf{SAT}}$.

In addition, a surprising fact is that Theorem 3.3 is a hardness result under *deterministic reductions*. A line of work [KC00, AHK17, MW17, HP15, HW16, AH17] showed that NP-hardness of MCSP or time-bounded Kolmogorov complexity under deterministic reductions implies circuit lower bounds (or a contradiction), which suggests that deterministic reductions are too strong a reducibility notion to prove hardness results for time-bounded Kolmogorov complexity. Theorem 3.3 is *the first hardness result* of polynomial-time-bounded Kolmogorov complexity that overcomes these "non-NP-hardness" results. Indeed, prior to this work, the only hardness results of Kolmogorov complexity under deterministic reductions we are aware of were (1) BPP-hardness of R_{K} under nonadaptive reductions [BFKL10], and (2) PSPACE-hardness of R_{K} under adaptive reductions [ABK+06b]. Their proof techniques do not work for exponential-time-bounded Kolmogorov complexity.[9]

A key to overcoming the non-NP-hardness barrier is that (1) MCSP is a problem about *sublinear-time*-bounded Kolmogorov complexity (whereas MINcKT is not), and that (2) one can compute a satisfying assignment in $\mathsf{P}^{\mathsf{SAT}}$. Both of these conditions are indispensable. Indeed, by adapting the proof techniques of [MW17, HW16], we show that it is impossible to improve the NP-hardness of $\mathrm{MINcKT}^{\mathsf{SAT}}$ to $\mathrm{MCSP}^{\mathsf{SAT}}$ or MINcKT without resolving the notorious open question that $\mathsf{EXP} \neq \mathsf{ZPP}$.

Proposition 3.4 (Tightness of Theorem 3.3)**.** *If either*

- $\mathrm{MCSP}^{\mathsf{SAT}}$ *is* NP*-hard under polynomial-time nonadaptive reductions, or*
- MINcKT *is* NP*-hard under polynomial-time nonadaptive reductions,*

then $\mathsf{EXP} \neq \mathsf{ZPP}$.

It is natural to ask whether a similar hardness result can be proved for an "unconditional" version $\mathrm{MINKT}^{\mathsf{SAT}}$ of $\mathrm{MINcKT}^{\mathsf{SAT}}$. We prove that $\mathrm{MINKT}^{\mathsf{SAT}}$ is average-case NP-hard in the following sense.

Theorem 3.5 (DistNP-hardness of $\mathrm{MINKT}^{\mathsf{SAT}}$)**.**

$$\textit{If } \mathrm{MINKT}^{\mathsf{SAT}} \in \mathsf{BPP}\textit{, then } \mathsf{DistNP} \subseteq \mathsf{HeurBPP}.$$

[8] One of the best hardness results of MCSP is the SZK-hardness shown by Allender and Das [AD17], where $\mathsf{SZK} \subseteq \mathsf{AM} \cap \mathsf{coAM}$.

[9] The hardness result of Buhrman, Fortnow, Koucký, and Loff [BFKL10] works for R_{K^t} where $t(n) = 2^{2^{2n}}$; however, it is impossible to prove a BPP-hardness result for $t(n) = 2^{n^2}$ without resolving $\mathsf{BPP} \neq \mathsf{EXP}$ because of a result of Buhrman and Mayordomo [BM97] (see [BFKL10]).

This suggests that the approach of [Hir18] for establishing an equivalence between worst-case and average-case complexity of NP is likely to be a "universal approach." Indeed, Theorem 3.5 implies that it is impossible to prove the implication that $\mathsf{NP} \not\subseteq \mathsf{BPP} \Rightarrow \mathsf{DistNP} \not\subseteq \mathsf{HeurBPP}$ (i.e., an equivalence between worst-case to average-case complexity of NP) without proving $\mathsf{NP} \not\subseteq \mathsf{BPP} \Rightarrow \mathrm{MINKT}^{\mathsf{SAT}} \notin \mathsf{BPP}$ (i.e., NP-hardness of $\mathrm{MINKT}^{\mathsf{SAT}}$ under non-black-box reductions). That is, proving NP-hardness of $\mathrm{MINKT}^{\mathsf{SAT}}$ is a prerequisite for establishing the equivalence between worst-case and average-case complexity of NP; Note that, conversely, proving NP-hardness of GapMINKT is sufficient for establishing a worst-case-and-average-case equivalence of NP [Hir18].

3.2 Hardness of Kolmogorov Complexity Under Nonadaptive Reductions

In the case of resource-unbounded Kolmogorov complexity, we can dispense with conditional Kolmogorov complexity using symmetry of information. Specifically, we prove that $\mathsf{S}_2^{\mathrm{p}}$ $\left(\supseteq \mathsf{P}^{\mathsf{NP}}\right)$ is reducible to computing Kolmogorov complexity via a quasipolynomial-time nonadaptive reduction.

Theorem 3.6. $\mathsf{S}_2^{\mathrm{p}} \subseteq \mathsf{quasiP}_{\|}^{\mathrm{K}(\text{-})}$, *where* $\mathrm{K}(\text{-})$ *is an oracle that answers* $\mathrm{K}(q)$ *for any query* $q \in \{0,1\}^*$ *and the class* $\mathsf{quasiP}_{\|}^{A}$ *denotes the class of problems nonadaptively reducible to* A *in quasipolynomial time.*

We briefly explain below the context of Theorem 3.6 and its importance. A line of work attempted to characterize BPP in terms of nonadaptive reductions to the set R_{K} of Kolmogorov-random strings. We summarize known lower bounds and upper bounds below.

Theorem 3.7 ([BFKL10, AFG13, CDE+14])**.**

$$\mathsf{BPP} \subseteq \bigcap_{U} \mathsf{P}_{\|}^{R_{\mathrm{K}_U}} \subseteq \bigcap_{U} \mathsf{P}_{\|}^{\mathrm{K}_U(\text{-})} \subseteq \mathsf{PSPACE},$$

where the intersections are taken over all prefix-free universal Turing machines U*.*

Allender [All12] and others [BFKL10, ADF+13, ABFL14, HK18] conjectured that the BPP lower bound for $\bigcap_U \mathsf{P}_{\|}^{R_{\mathrm{K}}}$ is tight. One of the main intuitions behind this conjecture is that, in the case of reductions to the set R_{K^t} of time-bounded Kolmogorov complexity, the upper bound can be improved to $\mathsf{PSPACE} \cap \mathsf{P/poly}$. Specifically,

Theorem 3.8 (Allender, Buhrman, Friedman, and Loff [ABFL14])**.** *For any time bound* t_0*, there exists a time bound* t *such that, for any language* $A \in \mathsf{DTIME}(t_0) \cap \mathsf{P}^{R_{\mathrm{K}^t}}$*, it holds that* $A \in \mathsf{P/poly}$*.*

This can be seen as a strong barrier result: It is impossible to reduce any problem in $\mathsf{EXP} \setminus \mathsf{P/poly}$ to R_{K^t} via a deterministic reduction for a sufficiently large t. In particular, in order to reduce, for example, NP to R_{K} via a deterministic reduction, we need to rely on a property of K that does not hold for K^t for a sufficiently large t.

In our previous work [Hir20], we essentially refuted the conjecture of Allender [All12], by showing that any sparse language in PH is quasipolynomial-time nonadaptively reducible to R_{K}; in particular, $\mathsf{BPP} \neq \bigcap_U \mathsf{P}_{\|}^{R_{\mathrm{K}_U}}$ unless $\mathsf{EXPH} = \mathsf{BPEXP}$. However, any sparse language is trivially in $\mathsf{P/poly}$, and thus, the reduction is subject to the barrier of Theorem 3.8.

Theorem 3.6 is the first hardness result under nonadaptive deterministic reductions that overcomes the barrier of Theorem 3.8 (unless $\mathsf{NP} \subseteq \mathsf{P/poly}$). A key for overcoming the barrier is the

usage of symmetry of information. Note that symmetry of information of Kolmogorov complexity does not necessarily hold in a time-bounded settings.[10]

3.3 Non-Trivial Satisfiability Algorithms and Immunity of Random Strings

Inspired by our efficient reductions, we obtain new lower bounds for the set of time-bounded Kolmogorov-random strings. For a complexity class $\mathfrak{C}$, a language $R \subseteq \{0,1\}^*$ is called $\mathfrak{C}$-*immune* (or immune to $\mathfrak{C}$) if R is infinite and there exists no infinite set $L \in \mathfrak{C}$ such that $L \subseteq R$. The notion of immunity is a much stronger lower bound than an average-case lower bound: The P-immunity of R implies that any errorless polynomial-time heuristic algorithm can solve R on only finitely many inputs in R. We herein show P-immunity of R_{K^t}.

Theorem 3.9. *R_{K^t} is P-immune for any super-polynomial t.*

Previously, Ko [Ko91] observed that R_{K^t} is immune to the class of P-printable languages; a P-printable language is, by the definition, contained in the set of nonrandom strings. Theorem 3.9 significantly strengthens this, and it resolves the open question mentioned in Section 1 (i.e., Theorem 3.9 implies Theorem 1.1).

We also show a lower bound against P-uniform ACC^0 for another variant of time-bounded Kolmogorov complexity. The *Levin's* Kt*-complexity* [Lev84] of a string $x \in \{0,1\}^*$ is defined as

$$\mathrm{Kt}(x) := \min\{\, |d| + \log t \mid U \text{ outputs } x \text{ in } t \text{ steps on input } d \,\}.$$

The set R_{Kt} of random strings with respect to Kt is known to be EXP-complete under $\mathsf{P/poly}$ reductions or NP-Turing reductions [ABK+06b]. It is an open question to prove a strong lower bound for R_{Kt}, such as $R_{\mathrm{Kt}} \notin \mathsf{P}$. For other variants of R_{Kt} such as R_{KNt} and R_{rKt} (which are NEXP-complete and BPEXP-complete variants of R_{Kt}, respectively), lower bounds against $\mathsf{NP} \cap \mathsf{coNP}$ and BPP, respectively, are known [AKRR11, AS12, Oli19]. The original question about R_{Kt}, however, remains elusive. We make a progress towards understanding the complexity of R_{Kt} by showing

Theorem 3.10. *R_{Kt} is (P-uniform ACC^0)-immune.*

These results are obtained by establishing a firm link between a non-trivial satisfiability algorithm and immunity. Here, by a *non-trivial satisfiability algorithm* for a circuit class $\mathfrak{C}$, we mean that an algorithm that takes an n-input size-s circuit $C \in \mathfrak{C}$ as input and outputs a satisfying assignment for C, if exists, in time $\mathsf{poly}(s) \cdot 2^{n-\omega(\log n)}$. In the celebrated work of Williams [Wil13, Wil14], he showed that a non-trivial satisfiability algorithm for $\mathfrak{C}$ implies $\mathsf{NEXP} \not\subseteq \mathfrak{C}$, and he presented a non-trivial satisfiability algorithm for non-uniform ACC^0, thereby establishing an *algorithmic approach* for circuit lower bounds.

How do we compare the approach of Kolmogorov mentioned in Section 1 with Williams' approach? At a first glance, these approaches seem to be quite orthogonal in that Williams' approach focuses on the satisfiability problem, which is a canonical NP-complete problem. Somewhat surprisingly, we make a connection among these different approaches: The existence of a non-trivial satisfiability algorithm is equivalent to proving a strong lower bound (immunity) for the set of random strings with respect to Levin's Kt-complexity.

[10] We mention that symmetry of information for resource-bounded Kolmogorov complexity studied in, e.g., [LW95, LR05] is not enough for our reductions to work. We also mention that symmetry of information provably does not hold for Kt [Ron04].

Theorem 3.11. *For any circuit class $\mathfrak{C}$, the following are equivalent.*

1. *There exists a non-trivial satisfiability algorithm for P-uniform $\mathfrak{C}$-circuits.*
2. *There exists a time-constructible function $s\colon \mathbb{N} \to \mathbb{N}$ satisfying $s(n) = n - \omega(\log n)$ such that the set of strings x such that $\mathrm{Kt}(x) \geq s(|x|)$ is immune to P-uniform $\mathfrak{C}$.*

3.4 Proof Techniques

$\mathsf{EXP}^{\mathsf{NP}}$-Hardness. We sketch a basic proof idea of Theorem 3.1 below. For simplicity, we provide a proof sketch for $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathsf{BPP}^{R_{\mathrm{K}}}$. To this end, we aim at constructing a pseudorandom generator whose security is based on a uniform hardness assumption of $\mathsf{EXP}^{\mathsf{NP}}$ (e.g., $\mathsf{EXP}^{\mathsf{NP}} \not\subseteq \mathsf{BPP}$).

Let $f = \{f_n : \{0,1\}^n \to \{0,1\}\}_{n\in\mathbb{N}}$ denote a computable function supposed to be hard. At the core of our results is the following well-known construction of a pseudorandom generator:

$$\mathrm{DP}_k^{f_n}(z_1, \cdots, z_k) = (z_1, \cdots, z_k, \widetilde{f_n}(z_1), \cdots, \widetilde{f_n}(z_k)).$$

Here, $\widetilde{f_n}$ is a version of f encoded by some appropriate locally-list-decodable error-correcting code. We refer $\mathrm{DP}_k^{f_n}$ as a *k-wise direct product generator*. This is a pseudorandom generator $\mathrm{DP}_k^{f_n} : \{0,1\}^d \to \{0,1\}^{d+k}$ that extends a seed of length $d = O(nk)$ to a pseudorandom sequence of length $d+k$. As mentioned in Section 2, the set R_{K} of Kolmogorov-random strings is a distinguisher for $\mathrm{DP}_k^{f_n}$ when k is large enough; indeed, for any seed $\bar{z}$ of length d,

$$\mathrm{K}(\mathrm{DP}_k^{f_n}(\bar{z})) \leq d + O(\log n) \ll d + k,$$

where we choose the parameter $k = O(\log n)$ large enough so that the last inequality holds. Therefore, R_{K} can distinguish the output distribution of $\mathrm{DP}_k^{f_n}(\text{-})$ from the uniform distribution of $(d+k)$-bit strings.

At a high level, our key idea is that the k-wise direct product generator is a pseudorandom generator construction with *very small advice complexity*; the advice complexity is so small that one can exhaustively search all advice strings efficiently. We explain the details below.

Recall the following standard security proof of the k-wise direct product generator $\mathrm{DP}_k^{f_n}$. By using a standard hybrid argument, it can be shown that, if there exists a distinguisher D for $\mathrm{DP}_k^{f_n}$, then there exists a small D-oracle circuit $\widetilde{C}^D$ that approximates $\widetilde{f_n}$. By using a locally-list-decoding algorithm, the circuit $\widetilde{C}^D$ can be converted to a small D-oracle circuit C^D that computes f_n on every input. Therefore, if f_n is indeed a hard function in the sense that no small circuit can compute f_n, then the circuit size of D must be large; in other words, $\mathrm{DP}_k^{f_n}$ is a pseudorandom generator secure against small circuits.

Unfortunately, the proof sketch above is not enough for obtaining a pseudorandom generator based on *uniform hardness assumptions*. Recall that our hardness assumption is that there exists a function $f \in \mathsf{EXP}^{\mathsf{NP}}$ such that f is not computable by efficient uniform algorithms (e.g., $f \notin \mathsf{BPP}$). In contrast, the security proof above produces a small circuit that computes a hard function f.

Impagliazzo and Wigderson [IW01] and Trevisan and Vadhan [TV07] constructed a pseudorandom generator based on uniform hardness assumptions of a problem in PSPACE, by exploiting downward self-reducibility and an instance checkability of $f \in \mathsf{PSPACE}$. Unfortunately, since any downward-self-reducible function f is in PSPACE, their proof ideas are not enough for overcoming the barrier of Gutfreund and Vadhan [GV08] in the black-box security reduction setting.

Our new insight is this: The "advice complexity" of the reconstruction procedure for the k-wise direct product is roughly at most k bits. In other words, given a distinguisher D for $\mathrm{DP}_k^{f_n}$, one can produce at most 2^k small D-oracle circuits $C_1, \cdots, C_{2^k}$, one of which is guaranteed to compute the hard function f_n. The following "Enumeration Lemma for Functions" encapsulates the key property of $\mathrm{DP}_k^{f_n}$.

Lemma 5.5 ((A corollary of) Enumeration Lemma for Functions)**.** *Let $f := \{f_n : \{0,1\}^n \to \{0,1\}\}_{n\in\mathbb{N}}$ be a computable family of functions. Then, there exists a randomized polynomial-time algorithm that, given 1^n as input, with high probability, outputs a list of R_{K}-oracle circuits $C_1^{R_{\mathrm{K}}}, \cdots, C_m^{R_{\mathrm{K}}}$ one of which computes f_n.*

It remains to compute f by a randomized polynomial-time algorithm with the help of oracles $C_1, \cdots, C_m$, one of which is guaranteed to compute f. This property is exactly captured by the notion of *(oracle) selector*, which was introduced in [Hir15]. The main result of [Hir15] is that there exists a randomized polynomial-time selector for any $\mathsf{EXP}^{\mathsf{NP}}$-complete problem. A *selector* for a function f is an efficient oracle algorithm that computes f, given oracle access to $C_1, \cdots, C_m$ one of which is guaranteed to compute f. The claim $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathsf{BPP}^{R_{\mathrm{K}}}$ follows by combining the Enumeration Lemma and the selector for an $\mathsf{EXP}^{\mathsf{NP}}$-complete problem.

Enumeration Lemma for Auxiliary-Input Functions. In order to prove the NP-hardness of $\mathrm{MINcKT}^{\mathsf{SAT}}$ (i.e., Theorem 3.3), we generalize the Enumeration Lemma for a function family $f = \{f_x : \{0,1\}^{\ell(|x|)} \to \{0,1\}\}_{x\in\{0,1\}^*}$ indexed by an auxiliary input $x \in \{0,1\}^*$: Assume that there exists an oracle D such that $D(x,\text{-})$ distinguishes the output distribution of $\mathrm{DP}_k^{f_x}(\text{-})$ from the uniform distribution. Then, by using the fact that the advice complexity of the direct product generator is small, given x as input, one can efficiently produce a list of D-oracle circuits one of which compute f_x. We call this "Enumeration Lemma for Auxiliary-Input Functions."

A key idea for obtaining NP-hardness of $\mathrm{MINcKT}^{\mathsf{SAT}}$ is as follows. For a formula φ and a string w, define an oracle D as $D(\varphi, w) := 1$ iff $\mathrm{K}^{t,\mathsf{SAT}}(w|\varphi) < |w| - 1$ for a sufficiently large polynomial t. Then, it can be shown that the oracle $D(\varphi,\text{-})$ distinguishes the output distribution of $\mathrm{DP}_k^{a_\varphi}(\text{-})$ from the uniform distribution for $k = O(1)$, where a_φ is a function that encodes a satisfying assignment of $\varphi \in \mathsf{SAT}$. By applying the Enumeration Lemma for Auxiliary-Input Functions, we obtain a randomized algorithm that enumerates a list of circuits one of which encodes a satisfying assignment for φ, which enables us to reduce SAT to $\mathrm{MINcKT}^{\mathsf{SAT}}$.

Hardness Under Deterministic Reductions. The outline above provides NP-hardness of $\mathrm{MINcKT}^{\mathsf{SAT}}$ under *randomized* reductions. By inspecting the proof carefully, the number of random bits used in the Enumeration Lemma for Auxiliary-Input Functions $f = \{f_x : \{0,1\}^{\ell(|x|)} \to \{0,1\}\}_{x\in\{0,1\}^*}$ can be reduced to $O(k \cdot \ell(|x|))$ bits, where k is the parameter of the direct product generator. Therefore, if the input length $\ell(|x|)$ of f_x is logarithmic and k is constant, one can simply exhaustively search random bits in polynomial time, which enables us to obtain NP-hardness of $\mathrm{MINcKT}^{\mathsf{SAT}}$ under deterministic reductions. To summarize the discussion above, a key to our hardness results under deterministic reductions is the following "Deterministic Enumeration Lemma for Strings."

Lemma 7.3 (Deterministic Enumeration Lemma for Strings)**.** *Let D be an oracle, $k, p\colon \mathbb{N} \to \mathbb{N}$ be polynomials. Let $y := \{y_x \in \{0,1\}^{p(|x|)}\}_{x\in\{0,1\}^*}$ be an indexed family of strings such that $D(x,\text{-})$ is a distinguisher for $\mathrm{DP}_{k(|x|)}^{y_x}(\text{-})$, for every $x \in \{0,1\}^*$, where we regard y_x as a function $y_x\colon \{0,1\}^{\lceil \log p(|x|)\rceil} \to \{0,1\}$. Then, there exists a deterministic $n^{O(k(n))}$-time oracle algorithm that,*

given $x \in \{0,1\}^*$ of length n as input and oracle access to D, outputs a list $Y \subseteq \{0,1\}^*$ of strings that contains y_x.

In the case of resource-unbounded Kolmogorov complexity, it is possible to define a distinguisher D without using conditional Kolmogorov complexity. Define D as $D(x,w) = 1$ if and only if $\mathrm{K}(x,w) \leq \mathrm{K}(x) + |w| - O(\log |x|)$ for $x, w \in \{0,1\}^*$. Using symmetry of information of Kolmogorov complexity, it can be shown that $D(x,\text{-})$ is a distinguisher for the k-wise direct product generator $\mathrm{DP}_k^{y_x}(\text{-})$, where $k = O(\log |x|)$ and $y = \{y_x \in \{0,1\}^{p(|x|)}\}_{x \in \{0,1\}^*}$ is an arbitrary computable indexed family of strings.

Proof Idea for the Unconditional Lower Bounds. We originally found a complicated proof outline for proving the lower bound $R_{\mathrm{K}^t} \notin \mathsf{P}$ of Theorem 1.1 for a sub-exponential time bound $t(n) = 2^{n^\epsilon}$ with a constant $\epsilon > 0$, based on our efficient reductions. The original idea was, using our hardness results under deterministic reductions, to obtain a contradiction with the average-case hierarchy theorem of [Wil83, GW00], which states that there exists a problem $L \in \mathsf{DTIME}(2^{n^\epsilon}) \setminus \mathsf{Heur\text{-}quasiP}$. Then we were able to significantly simplify the proof and strengthen the lower bound, based on the connection between non-trivial satisfiability algorithms and immunity. The main idea for proving $R_{\mathrm{K}^t} \notin \mathsf{P}$ is to have an advice string $a \in \{0,1\}^{n - O(\log n)}$, and to then try to solve the satisfiability problem for the remaining $O(\log n)$-bits input by an exhaustive search. This can be shown to be a "non-trivial satisfiability algorithm" with respect to the definition of K^t.

3.5 Summary and Concluding Remarks

We presented new hardness results of Kolmogorov complexity under *surprisingly efficient reductions*, which overcome several barrier results. We summarize hardness results and corresponding barriers below.

1. $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathsf{P}^{R_{\mathrm{K}}}$ (Theorem 3.1), which overcomes the barrier result of Gutfreund and Vadhan [GV08] (Theorem 2.3). A key to overcome the barrier is to consider a pseudorandom generator that stretches its seed by a small amount.

2. $\mathrm{MINcKT}^{\mathsf{SAT}}$ is NP-hard under deterministic reductions (Theorem 3.3), which overcomes the non-NP-hardness results of [KC00, AHK17, MW17, HP15, HW16, AH17] on MCSP. A key to overcome the barrier is that

   (a) $\mathrm{MINcKT}^{\mathsf{SAT}}$ is a problem about polynomial-time-bounded Kolmogorov complexity, whereas $\mathrm{MCSP}^{\mathsf{SAT}}$ is a problem about a *sublinear-time*-bounded Kolmogorov complexity, and that

   (b) one can compute a satisfying assignment in $\mathsf{P}^{\mathsf{SAT}}$.

   Note that both of these conditions are indispensable, as shown in Proposition 3.4.

3. $\mathsf{S}_2^{\mathrm{p}} \subseteq \mathsf{quasiP}_{\|}^{\mathrm{K}(\text{-})}$ (Theorem 3.6), which overcomes the $\mathsf{P/poly}$ barrier result of Allender, Buhrman, Friedman, and Loff [ABFL14] (Theorem 3.8). A key to overcome the barrier is to exploit symmetry of information of resource-unbounded Kolmogorov complexity.

It is worth emphasizing that our proof techniques provide nearly tight results from several perspectives. Theorem 3.2 is nearly tight in the sense that $\mathsf{EXP}^{\mathsf{NP}} \subseteq \bigcap_D \mathsf{BPP}^D \subseteq \mathsf{S}_2^{\mathrm{exp}}$, where the intersection is taken over all distinguishers D for the pseudorandom generator of Theorem 3.2.

Theorem 3.3 is tight in the sense of Proposition 3.4. These facts suggest that the direct product generator, one of the simplest constructions of a pseudorandom generator, is a key to understanding hardness of Kolmogorov complexity.

Organization

The rest of this paper is organized as follows. After reviewing some standard notions in Section 4, we prove the Enumeration Lemma in Section 5. Section 6 contains the applications of the Enumeration Lemma. We prove the Deterministic Enumeration Lemma in Section 7 and provide its application in Section 8. In Section 9, we establish the connection between a non-trivial satisfiability algorithm and immunity.

4 Preliminaries

Notation. A language $L \subseteq \{0,1\}^*$ is often identified with its characteristic function $L\colon \{0,1\}^* \to \{0,1\}$. We denote by $\mathsf{quasiP} := \mathsf{DTIME}(2^{\mathrm{polylog}(n)})$. Let $[n] := \{1,\dots,n\}$ for $n \in \mathbb{N}$. A function $t\colon \mathbb{N} \to \mathbb{N}$ is referred to as a *time bound* if t is non-decreasing and time-constructible. We often regard an integer $t \in \mathbb{N}$ as a constant function $t\colon \mathbb{N} \to \mathbb{N}$ always mapping to t.

For a Boolean function $f\colon \{0,1\}^n \to \{0,1\}$, we denote by $\mathtt{tt}(f)$ the truth table of f, i.e., the concatenation of $f(x)$ for all $x \in \{0,1\}^n$ in the lexicographical order. Conversely, for a string $y \in \{0,1\}^N$, the function $\mathtt{fn}(y)\colon \{0,1\}^{\lceil \log N \rceil} \to \{0,1\}$ is defined as $\mathtt{fn}(y)(i) :=$ (the ith bit of y) if $i \le N$ and $\mathtt{fn}(y)(i) := 0$ otherwise, where $i \in [2^{\lceil \log N \rceil}]$ is identified with a binary representation in $\{0,1\}^{\lceil \log N \rceil}$.

We often identify a circuit C with the function computed by C. For a circuit C, denote by $|C|$ the size of a circuit. For a string x and an oracle A, we denote by $\mathsf{size}^A(x)$ the size of a minimum A-oracle circuit that computes $\mathtt{fn}(x)$.

For strings $x, y \in \{0,1\}^*$ of the same length, we denote by $\mathsf{Dist}(x,y)$ the fraction of indices i such that $x_i \neq y_i$.

Kolmogorov Complexity. Let K_μ be any variant of Kolmogorov complexity. For a function $s\colon \mathbb{N} \to \mathbb{N}$, a string $x \in \{0,1\}^*$ is said to be $s(\text{-})$-*random* with respect to K_μ if $\mathrm{K}_\mu(x) \ge s(|x|)$. We denote by $R^s_{\mathrm{K}_\mu}$ the set of s-random strings with respect to K_μ, i.e., $R^s_{\mathrm{K}_\mu} := \{\, x \in \{0,1\}^* \mid \mathrm{K}_\mu(x) \ge s(|x|) \,\}$. Our results (except for Section 9) hold for any randomness threshold s such that $n - O(1) \le s(n) \le n-1$; for the sake of simplicity, we assume that $s(n) := n-1$ by default, and we drop the superscript s in this case, i.e., $R_{\mathrm{K}_\mu} := \{\, x \in \{0,1\}^* \mid \mathrm{K}_\mu(x) \ge |x| - 1 \,\}$. A simple counting argument shows:

Fact 4.1. $|\{0,1\}^n \setminus R_{\mathrm{K}_\mu}| \le 2^{n-1}$ *for any* $n \in \mathbb{N}$.

Proof. The number of non-$s(\text{-})$-random strings is bounded by the number of programs of length less than $s(n)$, which is at most $\sum_{i=0}^{s(n)-1} 2^i \le 2^{s(n)} = 2^{n-1}$. □

For a time bound $t\colon \mathbb{N} \to \mathbb{N}$ and an oracle A, the *conditional Kolmogorov complexity* of x given y is defined as

$$\mathrm{K}^{t,A}(x|y) := \min\{\, |d| \mid \ U^A(d,y) \text{ outputs } x \text{ in } t(|x|+|y|) \text{ steps.} \ \},$$

where U is a universal Turing machine. By default, we assume $A = \varnothing$ and omit the superscript A.

Pseudorandomness. Let $G\colon \{0,1\}^d \to \{0,1\}^m$ and $D\colon \{0,1\}^m \to \{0,1\}$ be functions. We say that D *distinguishes* the output distribution of $G(\text{-})$ from the uniform distribution if

$$\Pr_{z\sim\{0,1\}^d}[D(G(z)) = 1] - \Pr_{w\sim\{0,1\}^m}[D(w) = 1] \geq \frac{1}{2}.$$

Reduction. For an oracle complexity class $\mathfrak{C}$, we denote by $\mathfrak{C}_{\|}^R$ the class of languages that are reducible to R via a nonadaptive $\mathfrak{C}$-reduction, which is also known as a *truth-table reduction* in the case of $\mathfrak{C} = \mathsf{P}$. (Here, the subscript $\|$ stands for a *parallel* query.)

5 Randomized Enumeration Lemma for Functions

In this section, we prove the Enumeration Lemma by using a direct product generator. First, we abstract the property of a pseudorandom generator construction. Following Trevisan and Vadhan [TV07], we define the notion of black-box pseudorandom generator construction (in a slightly different way).

Definition 5.1 (Black-Box Pseudorandom Generator Construction [TV07])**.** *Let $G^{(\text{-})}\colon \{0,1\}^d \to \{0,1\}^m$ be an oracle algorithm that expects an oracle of the form $f\colon \{0,1\}^\ell \to \{0,1\}$.*

The oracle algorithm $G^{(\text{-})}$ is said to be a black-box pseudorandom generator construction *with advice complexity a if there exists a deterministic nonadaptive oracle algorithm* Rec *such that, for every function $f\colon \{0,1\}^\ell \to \{0,1\}$ and every function $D\colon \{0,1\}^m \to \{0,1\}$, if*

$$\Pr_{z\sim\{0,1\}^d}\left[D(G^f(z)) = 1\right] - \Pr_{w\sim\{0,1\}^m}[D(w) = 1] \geq \frac{1}{2},$$

then there exists an advice function $A\colon \{0,1\}^r \to \{0,1\}^a$ such that

$$\Pr_{s\sim\{0,1\}^r}\left[\forall x \in \{0,1\}^\ell,\ \mathrm{Rec}^D(x; s, A(s)) = f(x)\right] \geq \frac{1}{8(m-d)}.$$

The algorithm Rec *is referred to as a* reconstruction algorithm *that uses r random bits.*

We will use the following construction of a locally-list-decodable error-correcting code.

Lemma 5.2 (Locally-List-Decodable Error-Correcting Code; Sudan, Trevisan and Vadhan [STV01])**.** *There exists a function* Enc *such that:*

1. $\mathrm{Enc}(x; N, \epsilon)$ *outputs a string of length $\widehat{N} = \mathsf{poly}(N, 1/\epsilon)$ for any $x \in \{0,1\}^N$, and is computable in time $\mathsf{poly}(N, 1/\epsilon)$.*
2. *There exists a randomized nonadaptive oracle algorithm $\mathrm{Dec}^{(\text{-})}(N, \epsilon)$ running in time $\mathsf{poly}(\log N, 1/\epsilon)$ that, given oracle access to $r \in \{0,1\}^{\widehat{N}}$, outputs a list of deterministic oracle circuits $C_1, \cdots, C_L$ such that, for every $x \in \{0,1\}^N$ satisfying $\mathsf{Dist}(r, \mathrm{Enc}(x)) \leq \frac{1}{2} - \epsilon$, with high probability over the choice of coin flips of* Dec*, there exists an index $j \in [L]$ such that $C_j^{\mathtt{fn}(r)}$ computes the ith bit of x on input $i \in [N]$.*

Now we present a construction of a k-wise direct product generator.

Definition 5.3 (Direct Product Generator)**.** *Let $k \in \mathbb{N}$ be a parameter and $f\colon \{0,1\}^\ell \to \{0,1\}$ be a function. We denote by $\hat{f}\colon \{0,1\}^{\widehat{\ell}} \to \{0,1\}$ the function specified by the truth table $\mathrm{Enc}(\mathtt{tt}(f); 2^\ell, \epsilon := 1/4k) \in \{0,1\}^{2^{\widehat{\ell}}}$, where $\widehat{\ell} = O(\ell + \log k)$, and Enc is the locally-list-decodable error-correcting code of Lemma 5.2.*

The k-wise direct product generator $\mathrm{DP}^f_k\colon \{0,1\}^{\widehat{\ell}k} \to \{0,1\}^{\widehat{\ell}k+k}$ is defined as

$$\mathrm{DP}^f_k(x^1, \cdots, x^k) := (x^1, \cdots, x^k, \widehat{f}(x^1), \cdots, \widehat{f}(x^k))$$

for $(x^1, \cdots, x^k) \in \left(\{0,1\}^{\widehat{\ell}}\right)^k$.

By using a standard hybrid argument, it can be shown that DP^f_k is a black-box pseudorandom generator construction. The important property of this construction is that the advice complexity is quite small.

Theorem 5.4. *For a parameter $k \in \mathbb{N}$ and a function $f\colon \{0,1\}^\ell \to \{0,1\}$, the k-wise direct product generator $\mathrm{DP}^f_k\colon \{0,1\}^{k\widehat{\ell}} \to \{0,1\}^{k\widehat{\ell}+k}$ is a black-box pseudorandom generator construction such that*

1. *DP^f_k has advice complexity $a := k + O(\log(k\ell))$, and*
2. *DP^f_k has a $\mathsf{poly}(\ell, k)$-time reconstruction algorithm that uses $\mathsf{poly}(\ell, k)$ random bits.*

Proof. We define hybrid distributions between the output distribution of $\mathrm{DP}^f_k(\text{-})$ and the uniform distribution on $\{0,1\}^{k\widehat{\ell}+k}$. For any $i \in \{0, \cdots, k\}$, define the ith hybrid distribution H_i as the distribution of

$$(x^1, \cdots, x^k, \widehat{f}(x^1), \cdots, \widehat{f}(x^i), b_{i+1}, \cdots, b_k),$$

where $\bar{x} = (x^1, \cdots, x^k) \sim \left(\{0,1\}^{\widehat{\ell}}\right)^k$ and $b_{i+1}, \cdots, b_k \sim \{0,1\}$. By this definition, H_0 is identically distributed with the uniform distribution, and H_k is an identical distribution with $\mathrm{DP}^f_k(x^1, \cdots, x^k)$. Therefore, if

$$\Pr_{\bar{x}}\left[D(x^1, \cdots, x^k, \widehat{f}(x^1), \cdots, \widehat{f}(x^k)) = 1\right] - \Pr_{\bar{x},b}\left[D(x^1, \cdots, x^k, b_1, \cdots, b_k) = 1\right] \geq \frac{1}{2},$$

then

$$\mathop{\mathbb{E}}_{\substack{i\sim[k]\\ \bar{x},b}}\left[D(H_i) - D(H_{i-1})\right] \geq \frac{1}{2k}.$$

By an averaging argument, we obtain

$$\Pr_{\substack{i\sim[k],b\\ x^1,\cdots,x^{i-1},x^{i+1},\cdots,x^k}}\left[\mathop{\mathbb{E}}_{x^i\sim\{0,1\}^{\widehat{\ell}}}\left[D(H_i) - D(H_{i-1})\right] \geq \frac{1}{4k}\right] \geq \frac{1}{4k}. \tag{1}$$

Consider the following deterministic algorithm $\mathrm{Rec}^D_0(x; s, A_0(s))$: The coin flip sequence s is regarded as $i \sim [k]$, $x^1, \cdots, x^{i-1}, x^{i+1}, \cdots, x^k \sim \{0,1\}^{\widehat{\ell}}$, and $b \sim \{0,1\}^k$. We set $x^i := x$ and $A_0(s) := (\widehat{f}(x^1), \cdots, \widehat{f}(x^{i-1}), b_i, \cdots, b_k)$. Then, the output of Rec^D_0 is defined as $D(\bar{x}, A_0(s)) \oplus 1 \oplus b_i$.

By a standard calculation (see, e.g., [Vad12]), it follows from (1) that

$$\Pr_x\left[\mathrm{Rec}_0^D(x,s,A_0(s))=\widehat{f}(x)\right]\geq\frac{1}{2}+\frac{1}{4k}$$

with probability at least $1/4k$ over the random choice of $s=(i,x^{[k]\setminus\{i\}},b)$.

The final reconstruction algorithm $\mathrm{Rec}^D(x;s,s',A(s,s'))$ operates as follows. It simulates the decoding algorithm $\mathrm{Dec}^{(\cdot)}(2^\ell,1/4k)$ of Lemma 5.2 with oracle access to $O(\cdot):=\mathrm{Rec}_0^D(\cdot,s,A_0(s))$ using s' as randomness of Dec, and obtains a list of oracle circuits $C_1,\cdots,C_L$. We define the advice function A as $A(s,s'):=(A_0(s),j)$, where $j\in[L]$ is an index such that C_j^O computes f; by Lemma 5.2, with high probability over the choice of s', such an index j exists; j is chosen arbitrarily if there exists no such an index j. The algorithm Rec^D outputs $C_j^O(x)$.

Overall, the algorithm $\mathrm{Rec}^D(x;s,s',A(s,s'))$ computes $f(x)$ for every x with probability at least $\frac{1}{8k}$. The advice function A outputs at most $k+O(\log L)=k+O(\log(k\ell))$ bits. □

As a consequence of the black-box pseudorandom generator construction with small advice complexity, given a distinguisher D for DP_k^f, one can efficiently enumerate D-oracle circuits, one of which computes f. In a typical application of the following lemma, we will set $I:=\{1^n\}_{n\in\mathbb{N}}$.

Lemma 5.5 (Randomized Enumeration Lemma for Functions)**.** *Let D be an oracle, and let $k,\ell\colon\mathbb{N}\to\mathbb{N}$ be time-constructible functions. Let $f:=\{f_x:\{0,1\}^{\ell(|x|)}\to\{0,1\}\}_{x\in I}$ be an auxiliary-input function indexed by $I\subseteq\{0,1\}^*$. Assume that $D(x,\cdot)$ distinguishes the output distribution of $\mathrm{DP}_{k(|x|)}^{f_x}(\cdot)$ from the uniform distribution for every $x\in I$. Then, there exists a randomized $\mathsf{poly}(n,2^{k(n)},\ell(n))$-time algorithm that, given a string $x\in I$ of length n as input and oracle access to D, with high probability, outputs nonadaptive oracle circuits $C_1,\cdots,C_L$ such that there exists an index $i\in[L]$ such that $C_i^D(z)=f_x(z)$ for every $z\in\{0,1\}^{\ell(n)}$.*

Proof. Fix $n\in\mathbb{N}$ and an input $x\in I$ of length n. Let $k:=k(n)$ and $\ell=\ell(n)$. By the assumption, we have

$$\Pr_{\bar{z}}\left[D(x,\mathrm{DP}_k^{f_x}(\bar{z}))=1\right]-\Pr_w\left[D(x,w)=1\right]\geq\frac{1}{2}.$$

Let Rec be the reconstruction algorithm of the direct product generator from Theorem 5.4. By the property of the reconstruction algorithm (Definition 5.1), there exists an advice function $A\colon\{0,1\}^r\to\{0,1\}^a$ such that

$$\Pr_{s\sim\{0,1\}^r}\left[\forall z\in\{0,1\}^\ell,\ \mathrm{Rec}^{D(x,\cdot)}(z;s,A(s))=f_x(z)\right]\geq\frac{1}{8k},\tag{2}$$

where $a=k+O(\log(k\ell))$.

Here is a randomized algorithm E that enumerates candidate oracle circuits that compute f_x: Repeat the following $O(k)$ times. Pick $s\sim\{0,1\}^r$ randomly. For each $\alpha\in\{0,1\}^a$, output an oracle circuit $C_{s,\alpha}^O$ defined as $C_{s,\alpha}^O(z):=\mathrm{Rec}^{O(x,\cdot)}(z;s,\alpha)$ for $z\in\{0,1\}^\ell$. The running time of this algorithm is at most $O(k)\cdot\mathsf{poly}(n,\ell,2^a)\leq\mathsf{poly}(n,\ell,2^k)$.

We claim the correctness of the algorithm E. Consider one iteration of E. It follows from (2) that, with probability at least $\frac{1}{8k}$ over the choice of $s\sim\{0,1\}^r$, there exists $\alpha:=A(s)$ such that $\mathrm{Rec}^{D(x,\cdot)}(z;s,\alpha)=f_x(z)$ for every $z\in\{0,1\}^\ell$. Therefore, $C_{s,\alpha}^D(x)=\mathrm{Rec}^{D(x,\cdot)}(z;s,\alpha)=f_x(z)$ for every $z\in\{0,1\}^\ell$; that is, $C_{s,\alpha}$ is a D-oracle circuit that computes f_x correctly. Since E repeats this $O(k)$ times, with high probability, E enumerates at least one D-oracle circuit that computes f_x. □

6 Hardness Under Randomized Reductions

In this section, we provide applications of the Enumeration Lemma (Lemma 5.5). We first show that there exists an $\mathsf{EXP}^{\mathsf{NP}}$-computable pseudorandom generator whose security can be based on a uniform hardness assumption of $\mathsf{EXP}^{\mathsf{NP}}$.

Restatement of Theorem 3.2. For any constant $c \in \mathbb{N}$, there exists an $\mathsf{EXP}^{\mathsf{NP}}$-computable function $G = \{G_n : \{0,1\}^{n-c\log n} \to \{0,1\}^n\}_{n\in\mathbb{N}}$ such that $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathsf{BPP}^D$ for any distinguisher D for G.

For the proof of Theorem 3.2, we make use of a selector for an $\mathsf{EXP}^{\mathsf{NP}}$-complete problem.

Definition 6.1 (Selector; [Hir15])**.** *For a language L, an oracle algorithm M is referred to as a* selector for L *if, for any input x and any oracles A_0, A_1 one of which is equal to L, $M^{A_0,A_1}(x)$ computes $L(x)$ correctly.*

Lemma 6.2 ([Hir15])**.** *For any $\mathsf{EXP}^{\mathsf{NP}}$-complete problem L, there exists a randomized polynomial-time selector S for L. That is, $\Pr_S[S^{A_0,A_1}(x) = L(x)] \geq 1 - 2^{-|x|}$ for any input x and any oracles A_0, A_1 such that $L \in \{A_0, A_1\}$.*

Proof of Theorem 3.2. Take any paddable $\mathsf{EXP}^{\mathsf{NP}}$-complete problem L, and a randomized polynomial-time selector S for L. By padding queries of a selector, we may assume without loss of generality that, on inputs of length n, S makes queries of length exactly $\ell(n)$, where ℓ is some polynomial. As was shown in [Hir15], a selector S for L can be converted to another algorithm S' that computes L given oracle access to polynomially many oracles one of which is correct (instead of just two oracles).[11]

Define a family of functions $f := \{f_n : \{0,1\}^n \to \{0,1\}\}_{n\in\mathbb{N}}$ so that $f_n\colon \{0,1\}^n \to \{0,1\}$ is the characteristic function of $L \cap \{0,1\}^n$ for each $n \in \mathbb{N}$. Let $k(n) := c\log n$ for a large constant c, and define the pseudorandom generator $G_n := D^{f_n}_{k(n)}$. The pseudorandom generator G_n takes a seed of length $O(nk(n))$ and extends it by $k(n)$ bits.[12] Applying Lemma 5.5 to f, for any distinguisher for D, there exists a randomized polynomial-time algorithm E that, on input 1^n, outputs oracle circuits $C^D_1, \cdots, C^D_m$ one of which computes f_n, with high probability over the internal randomness of E.

The overall D-oracle algorithm for computing L is as follows. On input $x \in \{0,1\}^*$ of length n, run E on input $1^{\ell(n)}$ and obtain D-oracle circuits $C^D_1, \cdots, C^D_m$ one of which computes $f_{\ell(n)}$. Simulate the selector S' on input x with oracle access to $C^D_1, \cdots, C^D_m$, and output the answer of S'. □

Since R_{K} is a distinguisher for any computable hitting set generator, we obtain the following corollary.

Reminder of Theorem 3.1. $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathsf{P}^{R_{\mathrm{K}}}$.

[11] Proof Sketch: $S'^{A_1,\cdots,A_m}(x)$ simulates $S^{A_i,A_j}(x)$ for every $i, j \in [m]$, finds an index $i \in [m]$ and a bit $b \in \{0,1\}$ such that $S^{A_i,A_j}(x)$ outputs b for every $j \in [m]$, and outputs b.

[12] While this defines a pseudorandom generator that outputs $O(nk(n))+k(n)$, one can extend it to a pseudorandom generator $G'_m\colon \{0,1\}^{m-O(\log m)} \to \{0,1\}^m$ for every $m \in \mathbb{N}$.

Proof. It was shown in [ABK+06b] that $\mathsf{BPP}^{R_{\mathrm{K}}} = \mathsf{P}^{R_{\mathrm{K}}}$; thus it suffices to prove $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathsf{BPP}^{R_{\mathrm{K}}}$.

Let $s(n) = n - 3\log n$, and take the pseudorandom generator $G = \{G_n : \{0,1\}^{s(n)} \to \{0,1\}^n\}_{n\in\mathbb{N}}$ of Theorem 3.2. We claim that $\neg R_{\mathrm{K}}$ is a distinguisher for G. Indeed, since G is computable, for each $n \in \mathbb{N}$ and $z \in \{0,1\}^{s(n)}$, the string $G_n(z)$ can be described by $n \in \mathbb{N}$ and z; thus,

$$\mathrm{K}(G_n(z)) \leq 2\log n + s(n) + O(1) < n - 1,$$

where $2\log n + O(1)$ is an upper bound on a self-delimiting encoding of n. Therefore,

$$\Pr_{z\sim\{0,1\}^{s(n)}}[G_n(z) \notin R_{\mathrm{K}}] - \Pr_{w\sim\{0,1\}^n}[w \notin R_{\mathrm{K}}] \geq 1 - \frac{1}{2} = \frac{1}{2}.$$

By Theorem 3.2, we obtain that $\mathsf{EXP}^{\mathsf{NP}} \subseteq \mathsf{BPP}^{R_{\mathrm{K}}}$. □

The reason why the security reduction of Theorem 3.2 is adaptive is that the selector for an $\mathsf{EXP}^{\mathsf{NP}}$-complete problem is adaptive. In the case of randomized nonadaptive reductions, one can reduce NEXP to R_{K} based on the $\mathsf{MIP} = \mathsf{NEXP}$ theorem of Babai, Fortnow, and Lund [BFL91].

Theorem 6.3. *For any constant $c \in \mathbb{N}$, there exists an $\mathsf{EXP}^{\mathsf{NP}}$-computable function $G = \{G_n : \{0,1\}^{n-c\log n} \to \{0,1\}^n\}_{n\in\mathbb{N}}$ such that $\mathsf{NEXP} \subseteq \mathsf{BPP}^D_{\|}$ for any distinguisher D for G.*

For the proof of Theorem 6.3, we will use the following efficient proof system for NEXP. Moreover, a certificate for the proof system can be found in $\mathsf{EXP}^{\mathsf{NP}}$.

Lemma 6.4 (cf. [BFL91, BFLS91, BGH+05])**.** *For any $L \in \mathsf{NEXP}$, there exists a randomized nonadaptive oracle algorithm V such that, for any $x \in \{0,1\}^*$, the following holds.*

1. *If $x \in L$, then $\Pr_V\left[V^{O_x}(x) = 1\right] = 1$ for some oracle O_x. Moreover, given $x \in L$ and $y \in \{0,1\}^*$ as input, the output $O_x(y)$ of the oracle can be computed in $\mathsf{EXP}^{\mathsf{NP}}$.*
2. *If $x \notin L$, then $\Pr_V\left[V^O(x) = 1\right] \leq 2^{-|x|}$ for any oracle O.*

Proof of Theorem 6.3. We use the same construction G with Theorem 3.2. That is, for a paddable $\mathsf{EXP}^{\mathsf{NP}}$-complete function $f = \{f_n : \{0,1\}^n \to \{0,1\}\}_{n\in\mathbb{N}}$, we define $G_n := \mathrm{DP}^{f_n}_{k(n)}$ for $k(n) = c\log n$. Applying Lemma 5.5 to f, for any distinguisher for D, we obtain a randomized polynomial-time algorithm E that, on input 1^n, with high probability, outputs oracle circuits $C_1^D, \cdots, C_m^D$ one of which computes f_n. In light of this, our goal is to compute any $L \in \mathsf{NEXP}$ with the help of oracles $C_1^D, \cdots, C_m^D$ one of which computes an $\mathsf{EXP}^{\mathsf{NP}}$-complete problem.

Let $L \in \mathsf{NEXP}$, and take the efficient verifier V of Lemma 6.4. Let $\ell_V \colon \mathbb{N} \to \mathbb{N}$ be a polynomial that bounds the running time of V. Fix any input $x \in L$ of length $n \in \mathbb{N}$. Take an oracle O_x such that $V^{O_x}(x)$ accepts with probability 1. Since f is $\mathsf{EXP}^{\mathsf{NP}}$-complete, there exists a polynomial-time computable function R such that $O_x(y) = f_{\ell(n)}(R(x,y))$ for any string y of length at most $\ell_V(n)$, where $\ell\colon \mathbb{N} \to \mathbb{N}$ is some polynomial.

Consider the following nonadaptive D-oracle algorithm A for computing L. Let x be an input of length n. Run E on input $1^{\ell(n)}$ and obtain nonadaptive oracle circuits $C_1^D, \cdots, C_m^D$ one of which computes $f_{\ell(n)}$. Then, for each $i \in [m]$, check whether $V^{C_i^D(R(x,\text{-}))}(x) = 1$. Accept if this holds for some i; otherwise reject.

The correctness of A can be proved as follows. For any $x \notin L$, the algorithm A rejects with high probability because of the soundness of Lemma 6.4. Now consider any $x \in L$. By Lemma 5.5,

with high probability, there exists $i \in [m]$ such that $C_i^D(z) = f_{\ell(n)}(z)$ for any $z \in \{0,1\}^{\ell(n)}$. In particular, $C_i^D(R(x,y)) = f_{\ell(n)}(R(x,y)) = O_x(y)$ for any y of length at most $\ell_V(n)$; therefore, $V^{C_i^D(R(x,\text{-}))}(x) = V^{O_x(\text{-})}(x) = 1$. Thus, the algorithm A accepts with probability 1. □

Corollary 6.5. $\mathsf{NEXP} \subseteq \mathsf{BPP}_{\|}^{R_{\mathrm{K}}}$.

Proof Sketch. Since R_{K} is a distinguisher for any computable hitting set generator, the result follows from Theorem 6.3. □

Theorem 6.3 can be compared with the following limit of black-box security reductions of a pseudorandom generator (or more generally, a hitting set generator):

Theorem 6.6 (Hirahara and Watanabe [HW19])**.** *Let $G = \{G_n : \{0,1\}^{(1-\epsilon)n} \to \{0,1\}^n\}_{n\in\mathbb{N}}$ be any function, where $\epsilon > 0$ is a constant. Then,*

$$\bigcap_D \mathsf{BPP}_{\|}^D \subseteq \mathsf{NP/poly} \cap \mathsf{coNP/poly},$$

where the intersection is taken over all distinguishers D for a hitting set generator G.

That is, when the seed is extended by a factor of $\Omega(1)$, there exists no hitting set generator construction whose security is based on any problem outside $\mathsf{NP/poly} \cap \mathsf{coNP/poly}$ via a nonadaptive reduction. Theorem 6.3 bypasses this impossibility result by exploiting the fact that the pseudorandom generator construction extends a seed by a small amount.

Applying our proof techniques to the case of R_{K^t} with $t(n) = 2^{n^\epsilon}$ for a constant $\epsilon > 0$, we obtain the following EXP-completeness under efficient reductions.

Theorem 6.7. *Let $\epsilon > 0$ be any constant, and $t(n) := 2^{n^\epsilon}$. Then,*

- $\mathsf{EXP} \subseteq \mathsf{BPP}_{\|}^{R_{\mathrm{K}^t}}$, *and*
- $\mathsf{EXP} \subseteq \mathsf{ZPP}^{R_{\mathrm{K}^t}}$.

Allender et al. [ABK+06b] proved EXP-completeness of R_{K^t} under NP-Turing reductions or $\mathsf{P/poly}$ reductions; Theorem 6.7 improves the efficiency of these reductions.

We will use a randomized nonadaptive selector for an EXP-complete problem.

Lemma 6.8. *For any EXP-complete problem L, there exists a $\mathsf{BPP}_{\|}$-selector for L.*

Proof. There exists a nonadaptive *instance checker* for an EXP-complete problem in the sense of Blum and Kanan [BK95], which follows from an EXP-version of the efficient proof system of Lemma 6.4. As shown in [Hir15], a selector for L can be constructed from an instance checker for L. Since the existence of a selector is closed under polynomial-time reductions, there exists a randomized polynomial-time nonadaptive selector for an arbitrary EXP-complete problem. □

Proof of Theorem 6.7. Using the pseudorandom generator construction of [IW97], it was shown in [ABK+06b] that $\mathsf{BPP}^{R_{\mathrm{K}^t}} = \mathsf{ZPP}^{R_{\mathrm{K}^t}}$; thus it suffices to prove $\mathsf{EXP} \subseteq \mathsf{BPP}_{\|}^{R_{\mathrm{K}^t}}$.

Take an arbitrarily paddable EXP-complete problem $f = \{f_n : \{0,1\}^n \to \{0,1\}\}_{n\in\mathbb{N}}$. Let c be a constant such that f_n is computable in 2^{n^c}. By Lemma 6.8, there exists a randomized polynomial-time nonadaptive selector for f; thus, our goal is to use the Enumeration Lemma to enumerate R_{K^t}-oracle circuits one of which computes f. Let $\ell\colon \mathbb{N} \to \mathbb{N}$ be some polynomial chosen later. Define a padded version $f' = \{f'_n : \{0,1\}^{\ell(n)} \to \{0,1\}\}_{n\in\mathbb{N}}$ of f:

$$f'(x,y) := f(x),$$

where $x \in \{0,1\}^n$ and $y \in \{0,1\}^{\ell(n)-n}$. Let $k(n) := 3\log n$. We claim that the direct product generator $\mathrm{DP}^{f'_n}_{k(n)}\colon \{0,1\}^{\widehat{\ell}(n)k(n)} \to \{0,1\}^{\widehat{\ell}(n)k(n)+k(n)}$ can be distinguished by using the R_{K^t} oracle, where $\widehat{\ell}(n) \geq \ell(n)$. Indeed, for any seed $\bar{z}$ of length $\widehat{\ell}(n)k(n)$, $\mathrm{DP}_{k(n)}(\bar{z})$ can be described by the self-delimiting encoding of $n \in \mathbb{N}$ and $\bar{z}$ in time $2^{O(n^c)} \leq 2^{\ell(n)^\epsilon} = t(\ell(n))$ for a large polynomial $\ell(n) = O(n^{c/\epsilon})$. Therefore,

$$\mathrm{K}^t(\mathrm{DP}^{f'_n}_{k(n)}(\bar{z})) \leq |\bar{z}| + 2\log n + O(1) < \widehat{\ell}(n)k(n) + k(n) - 1,$$

which means that $\mathrm{DP}^{f'_n}_{k(n)}(\bar{z}) \notin R_{\mathrm{K}^t}$. By combining this with the counting argument of Fact 4.1, we conclude that

$$\Pr_{\bar{z}}\left[\mathrm{DP}^{f'_n}_{k(n)}(\bar{z}) \notin R_{\mathrm{K}^t}\right] - \Pr_{w}\left[w \notin R_{\mathrm{K}^t}\right] \geq 1 - \frac{1}{2} = \frac{1}{2}.$$

Applying Lemma 5.5 to R_{K^t}, we obtain a randomized polynomial-time algorithm that, on input 1^n, with high probability, enumerates R_{K^t}-oracle circuits one of which computes f'_n. Combining this algorithm with the nonadaptive selector for f, we obtain a randomized polynomial-time nonadaptive reduction from f to R_{K^t}. □

7 Deterministic Enumeration Lemma for Strings

The Randomized Enumeration Lemma for Functions (i.e., Lemma 5.5) provides a *randomized* algorithm that produces a list of R_{K}-oracle circuits that contains any function f of low Kolmogorov complexity $\mathrm{K}(f)$. The main reason why we need randomized algorithms in Lemma 5.5 is that the input length of a function f might be polynomially large.

In this section, we focus on the case when the input size of f is logarithmic. Identifying f with its truth table $\mathtt{tt}(f)$, we aim at computing a list of polynomial-length strings of low Kolmogorov complexity by a *deterministic* algorithm, which leads us to Deterministic Enumeration Lemma for Strings.

To this end, we replace the usage of the *randomized*-locally-list-decodable error-correcting code with a *deterministic*-list-decodable error-correcting code, which can be obtained by concatenating the Reed-Solomon code with the Hadamard code.

Lemma 7.1 (List-Decodable Error-Correcting Code; cf. [Sud97, KS99])**.** *There exists a function* Enc *such that:*

1. $\mathrm{Enc}(x; N, \epsilon)$ *outputs a string of length* $\widehat{N} = \mathsf{poly}(N, 1/\epsilon)$ *for any* $x \in \{0,1\}^N$*, and is computable in time* $\mathsf{poly}(N, 1/\epsilon)$*.*

2. *There exists a deterministic algorithm* $\mathrm{Dec}(\text{-}; N, \epsilon)$ *running in time* $\mathsf{poly}(N, 1/\epsilon)$ *such that, given any* $r \in \{0,1\}^{\widehat{N}}$*, outputs a list of all the strings* $x \in \{0,1\}^N$ *such that* $\mathsf{Dist}(r, \mathrm{Enc}(x; N, \epsilon)) \leq \frac{1}{2} - \epsilon$.

Using this error-correcting code, we provide a black-box pseudorandom generator construction whose randomness complexity is small; this will be a key property for obtaining the Deterministic Enumeration Lemma (by simply exhaustively searching all the random bits).

Theorem 7.2. *Let* $f\colon \{0,1\}^\ell \to \{0,1\}$ *be a function, and* $k \in \mathbb{N}$*. There exists a black-box pseudorandom generator construction* $G_k^f\colon \{0,1\}^{k\widehat{\ell}} \to \{0,1\}^{k\widehat{\ell}+k}$ *for some* $\widehat{\ell} = O(\ell + \log k)$ *such that*

1. $G_k^f(\bar{x})$ *is computable in time* $\mathsf{poly}(2^\ell, k)$ *on input* $\bar{x} \in \{0,1\}^{k\widehat{\ell}}$ *and* f*.*
2. *There exists a* $\mathsf{poly}(2^\ell, k)$*-time reconstruction algorithm for* G_k^f *that uses* $O(k\ell + k\log k)$ *random bits.*
3. G_k^f *has advice complexity* $a := k + O(\log k + \ell)$*.*

Proof. The pseudorandom generator construction G_k is the same with Definition 5.3, except that we use the error-correcting code of Lemma 7.1.

Specifically, let Enc denote the error-correcting code of Lemma 7.1, and let $\hat{f}\colon \{0,1\}^{\widehat{\ell}} \to \{0,1\}$ be the function specified by the truth table $\mathrm{Enc}(f; 2^\ell, \epsilon := 1/4k) \in \{0,1\}^{2^{\widehat{\ell}}}$, where $\widehat{\ell} = O(\ell + \log k)$. The pseudorandom generator construction $G_k^f\colon \{0,1\}^{\widehat{\ell}k} \to \{0,1\}^{\widehat{\ell}k+k}$ is defined as

$$G_k^f(x^1, \cdots, x^k) := (x^1, \cdots, x^k, \widehat{f}(x^1), \cdots, \widehat{f}(x^k))$$

for $(x^1, \cdots, x^k) \in \left(\{0,1\}^{\widehat{\ell}}\right)^k$.

We prove that G_k^f is a black-box pseudorandom generator construction with the claimed parameters, by using an almost identical proof with Theorem 5.4. Assume that D satisfies

$$\Pr_{\bar{x}}\left[D(x^1, \cdots, x^k, \widehat{f}(x^1), \cdots, \widehat{f}(x^k)) = 1\right] - \Pr_{\bar{x},b}\left[D(x^1, \cdots, x^k, b_1, \cdots, b_k) = 1\right] \geq \frac{1}{2}.$$

By using the same argument with Theorem 5.4, there exists an efficient oracle algorithm R_0 and an advice function A_0 such that

$$\Pr_x\left[R_0^D(x, s, A_0(s)) = \widehat{f}(x)\right] \geq \frac{1}{2} + \frac{1}{4k}$$

with probability at least $1/4k$ over the random choice of $s = (i, x^{[k]\setminus\{i\}}, b) \sim [k] \times \left(\{0,1\}^{\widehat{\ell}}\right)^{[k]\setminus\{i\}} \times \{0,1\}^k$. By evaluating $R_0^D(x, s, A_0(s))$ for every $x \in \{0,1\}^{\widehat{\ell}}$, we obtain a string $f' \in \{0,1\}^{2^{\widehat{\ell}}}$ that encodes a function that agrees with $\widehat{f}$ on at least a $1/2 + 1/4k$ fraction of inputs.

The final reconstruction algorithm $R^D(x; s, A(s))$ runs the decoding algorithm $\mathrm{Dec}(f'; 2^\ell, 1/4k)$ of Lemma 7.1, and obtains a list of strings $f_1, \cdots, f_L$. We define the advice function A as $A(s) := (A_0(s), j)$, where $j \in [L]$ is an index such that f_j coincides with the truth table of f. The algorithm $R^D(x; s, A(s))$ outputs the xth position of f_j.

The number $|s|$ of random bits used by the reconstruction algorithm is at most $O(\log k + \widehat{\ell}(k-1) + k) = O(k\ell + k\log k)$. The advice complexity is at most $|A_0(\text{-})| + \log L = k + O(\ell + \log k)$. □

Lemma 7.3 (Deterministic Enumeration Lemma for Strings)**.** *Let D be an oracle, $k, p\colon \mathbb{N} \to \mathbb{N}$ be polynomials. Let $y := \{y_x \in \{0,1\}^{p(|x|)}\}_{x\in\{0,1\}^*}$ be an indexed family of strings such that $D(x, \text{-})$ distinguishes the output distribution of $G^{\mathtt{fn}(y_x)}_{k(|x|)}(\text{-})$ from the uniform distribution for every $x \in \{0,1\}^*$, where $G^{(\text{-})}$ is the black-box pseudorandom generator construction of Theorem 7.2. Then, there exists a deterministic $n^{O(k(n))}$-time nonadaptive oracle algorithm that, given $x \in \{0,1\}^*$ of length n as input and oracle access to D, outputs a list $Y \subseteq \{0,1\}^*$ of strings such that $y_x \in Y$.*

Proof. Fix any $n \in \mathbb{N}$ and any input $x \in \{0,1\}^n$ of length n. Let $k := k(n)$. By the assumption, we have

$$\Pr_z\left[D\left(x, G_k^{\mathtt{fn}(y_x)}(z)\right) = 1\right] - \Pr_w\left[D(x,w) = 1\right] \geq \frac{1}{2}.$$

Let Rec be the reconstruction algorithm of $G^{(\text{-})}$ from Theorem 7.2. By the property of the reconstruction algorithm (Definition 5.1), there exists an advice function $A\colon \{0,1\}^r \to \{0,1\}^a$ such that

$$\Pr_{s\sim\{0,1\}^r}\left[\forall i \in \{0,1\}^\ell,\ \mathrm{Rec}^D(i; s, A(s)) = \mathtt{fn}(y_x)(i)\right] \geq \frac{1}{8k}, \tag{3}$$

where $\ell := \lceil \log |y_x| \rceil = O(\log n)$, $a = k + O(\log k + \ell)$, and $r = O(k\ell + k \log k)$.

Here is a D-oracle algorithm E that enumerates candidate strings of y_x: On input x of length $n \in \mathbb{N}$, for each random bit $s \in \{0,1\}^s$ and each advice string $\alpha \in \{0,1\}^a$, output the string obtained by concatenating $\mathrm{Rec}^D(i; s, \alpha)$ for each $i \in [p(n)]$. Obverse that the running time of this algorithm is at most $\mathsf{poly}(n, 2^r, 2^a) = \mathsf{poly}(n^{k(n)})$.

It is easy to see that y_x is contained in the list of strings enumerated by the algorithm E. Indeed, it follows from (3) that there exists a random bit $s \in \{0,1\}^s$ such that $\mathrm{Rec}^D(i; s, A(s))$ is equal to the ith bit of y_x for any $i \in [p(n)]$. Therefore, for $\alpha := A(s)$, the output of E coincides with y_x. □

8 Hardness Under Deterministic Reductions

Now we apply the Deterministic Enumeration Lemma (Lemma 7.3) in order to obtain hardness of Kolmogorov complexity under deterministic reductions. In Section 8.1, we present NP-hardness of $\mathrm{MINcKT}^{\mathsf{SAT}}$ along with its tightness. In Section 8.2, we make use of symmetry of information and show how to eliminate the usage of conditional Kolmogorov complexity; In Section 8.3, we prove DistNP-hardness of $\mathrm{MINKT}^{\mathsf{SAT}}$, which is another approach for eliminating conditional Kolmogorov complexity.

8.1 Hardness of MINcKT with SAT Oracle

We extend the definition of MINKT (Ko [Ko91]) to a conditional Kolmogorov complexity version. As mentioned in Section 3, we assume that a time bound t is sufficiently large, as otherwise the definition of t-time-bounded Kolmogorov complexity might depend on a particular choice of a universal Turing machine. In the following definition, we introduce a parameter t_0, which specifies a minimum time bound for any query to MINcKT.

Definition 8.1 (Minimum Conditional Time-Bounded Kolmogorov Complexity Problem)**.** *For a time bound $t_0\colon \mathbb{N} \to \mathbb{N}$ and an oracle A, $\mathrm{MINcKT}^A_{t_0}$ is defined as follows:*

$$\mathrm{MINcKT}^A_{t_0} := \{\, (x, y, 1^s, 1^t) \mid \ \mathrm{K}^{t,A}(x|y) \leq s \text{ and } t \geq t_0(|x| + |y|) \,\}.$$

Our hardness of $\mathrm{MINcKT}^{\mathsf{SAT}}_t$ is "stable" with respect to the choice of a time bound t: For any polynomial time bound t, P^{NP} is reducible to $\mathrm{MINcKT}^{\mathsf{SAT}}_t$ via a nonadaptive reduction. (In fact, our hardness results hold for a version of $\mathrm{GapMINcKT}^{\mathsf{SAT}}$ where there is a large gap between the time bounds of YES and NO instances.)

Restatement of Theorem 3.3. For any polynomial t_0, $\mathrm{MINcKT}^{\mathsf{SAT}}_{t_0}$ is P^{NP}-hard under polynomial-time nonadaptive reductions.

We will reduce the following canonical P^{NP}-complete problem to $\mathrm{MINcKT}^{\mathsf{SAT}}_t$

Lemma 8.2 (Lexicographically First Satisfying Assignment; Krentel [Kre88])**.** *For a formula φ of n variables, denote by $a_\varphi \in \{0,1\}^*$ its lexicographically first satisfying assignment; if φ is not satisfiable, define $a_\varphi := 1^n$. The following problem, denoted by* LexSat*, is P^{NP}-complete.*

$$\textsc{LexSat} := \{\, (\varphi, i) \mid \textit{the } i\textit{th bit of } a_\varphi \textit{ is } 1 \,\}.$$

Proof of Theorem 3.3. By Lemma 8.2, it suffices to prove that there exists a polynomial-time nonadaptive oracle algorithm that computes a_φ, given φ as input and oracle access to $\mathrm{MINcKT}^{\mathsf{SAT}}_{t_0}$. The idea is to enumerate a list of candidates for a_φ by using the Deterministic Enumeration Lemma. To this end, we claim that the $\mathrm{MINcKT}^{\mathsf{SAT}}_{t_0}$ oracle gives rise to a distinguisher $D(\varphi, \text{-})$ for the k-wise direct product generator $G_k^{\mathtt{fn}(a_\varphi)}$, where k is a constant. Details follow.

Let k be some parameter chosen later. Let $z \in \{0,1\}^d$ be any seed of $G_k^{\mathtt{fn}(a_\varphi)} \colon \{0,1\}^d \to \{0,1\}^{d+k}$, where $G_k^{(\text{-})}$ is the black-box pseudorandom generator construction of Theorem 7.2. We claim that $\mathrm{K}^{t,\mathsf{SAT}}(G_k^{\mathtt{fn}(a_\varphi)}(z) \mid \varphi)$ is relatively small for a large time bound t. Observe that $\mathrm{K}^{t,\mathsf{SAT}}(a_\varphi|\varphi) = O(1)$ for some polynomial t. Indeed, by using the SAT oracle and the downward self-reducibility of SAT, one can compute the lexicographically first satisfying assignment a_φ of φ in $\mathsf{P}^{\mathsf{SAT}}$. The string $G_k^{\mathtt{fn}(a_\varphi)}(z)$ can be described by using the integer $k \in \mathbb{N}$, the seed z, and the program of size $\mathrm{K}^{t,\mathsf{SAT}}(a_\varphi|\varphi) = O(1)$; therefore, for a sufficiently large polynomial $t \geq t_0$,

$$\mathrm{K}^{t,\mathsf{SAT}}(G_k^{\mathtt{fn}(a_\varphi)}(z) \mid \varphi) \leq d + O(\log k).$$

We choose a constant k large enough so that this is bounded above by $d + k - 2$. Define an oracle D so that $D(\varphi, y) := 1$ if and only if $\mathrm{K}^{t,\mathsf{SAT}}(y|\varphi) \leq |y| - 2$. Then we have $D(\varphi, G_k^{\mathtt{fn}(a_\varphi)}(z)) = 1$ for any seed $z \in \{0,1\}^d$. Note that D is reducible to $\mathrm{MINcKT}^{\mathsf{SAT}}_{t_0}$ because $t \geq t_0$.

On the other hand, for a string $w \sim \{0,1\}^{d+k}$ uniformly chosen at random, we have $D(\varphi, w) = 0$ with probability at least $\frac{1}{2}$ by Fact 4.1. Therefore,

$$\Pr_z\left[D\left(\varphi, G_k^{\mathtt{fn}(a_\varphi)}(z)\right) = 1\right] - \Pr_w\left[D(\varphi, w) = 1\right] \geq \frac{1}{2}.$$

By Lemma 7.3, given nonadaptive oracle access to $\mathrm{MINcKT}^{\mathsf{SAT}}_{t_0}$, one can output a list Y_φ of strings such that $a_\varphi \in Y_\varphi$ in time $|\varphi|^{O(k)} = \mathsf{poly}(|\varphi|)$.

The algorithm for computing a_φ with the $\mathrm{MINcKT}^{\mathsf{SAT}}_{t_0}$ oracle is as follows: On input φ, compute Y_φ and output $a' := \min\{\, a \in Y_\varphi \mid \varphi(a) = 1 \,\}$. It is easy to see that the output a' is equal to a_φ using the minimality of a_φ and $a_\varphi \in Y_\varphi$. □

A natural question is whether Theorem 3.3 can be improved to MINcKT without any oracle. We show that this is impossible without proving $\mathsf{EXP} \neq \mathsf{ZPP}$, based on the proof techniques of [MW17, HW16].

Proposition 8.3. *There exists a polynomial t_0 such that, if MINcKT_{t_0} is ZPP-hard under polynomial-time nonadaptive reductions, then $\mathsf{EXP} \neq \mathsf{ZPP}$.*

Proof. Assume, towards a contradiction, that $\mathsf{EXP} = \mathsf{ZPP}$. Then we also have $\mathsf{EXP} \subseteq \mathsf{P/poly}$. Under this assumption, Allender, Koucký, Ronneburger, and Roy [AKRR11] showed that $\mathsf{size}(x) \leq \mathsf{poly}(\mathrm{Kt}(x), \log|x|)$ for every string x.

Fix any language $L \in \mathsf{ZPEXP}$. We claim that $L \in \mathsf{EXP}$. Let $c \geq 1$ be a constant such that 2^{n^c} is an upper bound on the running time of a ZPEXP algorithm for L. Let $L' := \{\, x10^{2^{|x|^c}} \mid x \in L \,\}$ be a padded version of L. Since $L' \in \mathsf{ZPP}$, by the assumption, L' is reducible to MINcKT_{t_0} via some polynomial-time nonadaptive reduction R.

Fix any input $y := x10^{2^{|x|^c}}$ of L'. Let $q = (q_0, q_1, 1^s, 1^t)$ be an arbitrary query of the reduction R on input y. Since R is a nonadaptive reduction, any query q can be described by the string x (whose length is $O(\log|y|)$) and the index of q in polynomial time; thus, $\mathrm{Kt}(q) \leq O(\log|y|)$, and hence $\mathsf{size}(q) \leq \mathrm{polylog}(|y|)$. Since a circuit can be evaluated efficiently, we have $\mathrm{K}^{t_0'}(q) \leq \widetilde{O}(\mathsf{size}(q)) \leq \mathrm{polylog}(|y|)$ for some polynomial t_0'. In particular, $\mathrm{K}^{t_0}(q_0|q_1) \leq \mathrm{K}^{t_0'}(q) + O(\log|q|) \leq \mathrm{polylog}(|y|)$ for any sufficiently large polynomial t_0.

We now present a quasipolynomial-time algorithm for deciding L' by simulating the reduction R. Let $q = (q_0, q_1, 1^s, 1^t)$ be any query of R. If $t < t_0(|q_0| + |q_1|)$, by the definition of MINcKT_{t_0}, the query can be answered with "No." Otherwise, in order to answer the query q, it suffices to perform an exhaustive search of a description d of length at most $\mathrm{polylog}(|y|)$ such that $U(d, q_1)$ outputs q_0 in time t. Indeed, since $\mathrm{K}^t(q_0|q_1) \leq \mathrm{polylog}(|y|)$, this exhaustive search correctly finds the conditional Kolmogorov complexity $\mathrm{K}^t(q_0|q_1)$. Therefore, $L' \in \mathsf{quasiP}$. It follows that L can be solved in time $2^{\mathrm{polylog}(2^{|x|^c})} = 2^{|x|^{O(1)}}$ on input x; that is, $L \in \mathsf{EXP}$.

Since L is an arbitrary language in ZPEXP, we obtain that $\mathsf{ZPEXP} = \mathsf{EXP} = \mathsf{ZPP}$, which is a contradiction. (Indeed, since $\mathsf{EXP} = \mathsf{ZPP}$, we also have $\mathsf{EEXP} = \mathsf{ZPEXP}$ by a padding argument; thus $\mathsf{EEXP} = \mathsf{ZPEXP} = \mathsf{ZPP} = \mathsf{EXP}$, which contradicts the time hierarchy theorem.) □

By using an almost identical proof, we show that $\mathrm{MCSP}^{\mathsf{SAT}}$ cannot be NP-hard under polynomial-time nonadaptive reductions without proving $\mathsf{EXP} \neq \mathsf{ZPP}$. Recall the definition of the Minimum A-Oracle Circuit Size Problem ([KC00, AHK17]). The Minimum A-Oracle Circuit Size Problem is defined as

$$\mathrm{MCSP}^A := \{\, (f, s) \in \{0,1\}^* \times \mathbb{N} \mid \mathsf{size}^A(f) \leq s \,\}.$$

Proposition 8.4. *Let $A \in \mathsf{EXP}$ be an arbitrary oracle. If MCSP^A is NP-hard under polynomial-time nonadaptive reductions, then $\mathsf{EXP} \neq \mathsf{ZPP}$.*

Proof Sketch. Assume that $\mathsf{EXP} = \mathsf{ZPP}$. Following the proof of Proposition 8.3, we take any language $L \in \mathsf{ZPEXP}$, and consider a padded version $L' \in \mathsf{ZPP}$, where L' is a sparse language. It suffices to claim that $L' \in \mathsf{quasiP}$ by simulating a polynomial-time nonadaptive reduction R to MCSP^A.

Fix any input y of length $n \in \mathbb{N}$ for L'. Consider any query $q = (f, s)$ of the reduction R on the input y. Since L' is sparse and R is nonadaptive, it follows that $\mathrm{Kt}(f) \leq O(\log n)$. Under the assumption that $\mathsf{EXP} = \mathsf{ZPP} \subseteq \mathsf{P/poly}$, we have $\mathsf{size}(f) \leq \mathsf{poly}(\mathrm{Kt}(f), \log|f|) \leq \mathrm{polylog}(n)$. In particular, $\mathsf{size}^A(f) \leq \mathsf{size}(f) \leq s(n)$ for some bound $s(n) = \mathrm{polylog}(n)$. In order to answer a query q of the reduction R to MCSP^A, one can exhaustively search all the A-oracle circuits of size at most $s(n)$. Note that one can evaluate any A-oracle circuit of size $s(n)$ in time $2^{s(n)^{O(1)}}$, because the

length of any query of an A-oracle circuit is at most $s(n)$ and $A \in \mathsf{EXP}$. Therefore, the reduction R can be simulated in quasipolynomial time; hence, $L' \in \mathsf{quasiP}$. □

Summarizing Proposition 8.3 and Proposition 8.4, we obtain the following.

Restatement of Proposition 3.4. There exists a polynomial t_0 such that, if either

- $\mathrm{MCSP}^{\mathsf{SAT}}$ is NP-hard under polynomial-time nonadaptive reductions, or
- MINcKT_{t_0} is NP-hard under polynomial-time nonadaptive reductions,

then $\mathsf{EXP} \neq \mathsf{ZPP}$.

8.2 Symmetry of Information and Hardness of Kolmogorov Complexity

In the case of resource-unbounded Kolmogorov complexity, we can exploit symmetry of information and dispense with the conditional Kolmogorov complexity.

Lemma 8.5 (Symmetry of Information; cf. [LV08])**.** *For any $x, y \in \{0,1\}^*$, it holds that*

$$\mathrm{K}(x,y) \geq \mathrm{K}(x) + \mathrm{K}(y|x) - O(\log \mathrm{K}(x,y)).$$

Reminder of Theorem 3.6. $\mathsf{S}_2^{\mathrm{p}} \subseteq \mathsf{quasiP}_{\|}^{\mathrm{K}(\text{-})}$.

Proof. Let $L \in \mathsf{S}_2^{\mathrm{p}}$, and V be an $\mathsf{S}_2^{\mathrm{p}}$-type verifier for L. That is, V is a polynomial-time algorithm such that, for some polynomial p, for every input $x \in \{0,1\}^*$ of length $n \in \mathbb{N}$,[13]

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

For each $x \in \{0,1\}^*$, denote by y_x the lexicographically first string such that $V(x,y_x,z) = L(x)$ for all $z \in \{0,1\}^{p(n)}$. Similarly, denote by z_x the lexicographically first string such that $V(x,y,z_x) = L(x)$ for all $y \in \{0,1\}^{p(n)}$. We will enumerate candidate strings for y_x (and z_x) by using Lemma 7.3. In order to apply Lemma 7.3, we construct a distinguisher $D(x,\text{-})$ for $G_k^{\mathtt{fn}(y_x)}(\text{-})$ for some $k = O(\log n)$. Let $G_k^{\mathtt{fn}(y_x)} \colon \{0,1\}^d \to \{0,1\}^{d+k}$ be the black-box pseudorandom generator construction of Theorem 7.2.

Fix any input $x \in \{0,1\}^*$ of length $n \in \mathbb{N}$. Let $c_0 \in \mathbb{N}$ be some absolute constant chosen later. We define D so that $D(x,w) := 1$ if and only if $\mathrm{K}(x,w) \leq \mathrm{K}(x) + d + c_0 \cdot \log|x|$ for $w \in \{0,1\}^{d+k}$. Note that D is efficiently computable given nonadaptive oracle access to the K(-) oracle.

Observe that y_x is computable; thus $\mathrm{K}(y_x|x) = O(1)$. For any seed $z \in \{0,1\}^d$ of $G_k^{\mathtt{fn}(y_x)}$, the string $(x, G_k^{\mathtt{fn}(y_x)}(z))$ can be described by the shortest program for x and its length, the seed $z \in \{0,1\}^d$, and the program of size $\mathrm{K}(y_x|x)$; therefore,

$$\mathrm{K}(x, G_k^{\mathtt{fn}(y_x)}(z)) \leq \mathrm{K}(x) + d + c_0 \log n$$

[13] We follow the definition of $\mathsf{S}_2^{\mathrm{p}}$ given by [Can96], which is equivalent to that of [RS98] (cf. [Cai07]).

for some universal constant c_0, and hence $D(x, G_k^{\mathtt{fn}(y_x)}(z)) = 1$.

Now consider $\mathrm{K}(x, w)$ for a string $w \sim \{0,1\}^{d+k}$ chosen uniformly at random. By symmetry of information (Lemma 8.5), we obtain

$$\mathrm{K}(x, w) \geq \mathrm{K}(x) + \mathrm{K}(w|x) - O(\log n).$$

By the counting argument of Fact 4.1, we have $\mathrm{K}(w|x) \geq |w| - 1$ with probability at least $\frac{1}{2}$. Therefore,

$$\Pr_{w \sim \{0,1\}^{d+k}}[\mathrm{K}(x, w) \geq \mathrm{K}(x) + d + k - O(\log n)] \geq \frac{1}{2}.$$

This means that we can choose $k = O(\log n)$ large enough so that $\Pr_w[D(x, w) = 0] \geq \frac{1}{2}$.

To summarize, we have established that

$$\Pr_z[D(x, G_k^{\mathtt{fn}(y_x)}(z)) = 1] - \Pr_w[D(x, w) = 1] \geq \frac{1}{2}.$$

Applying Lemma 7.3, on input $x \in \{0,1\}^*$, one can output a list Y_x of strings such that $y_x \in Y_x$ in time $n^{O(k)} = n^{O(\log n)}$. Similarly, one can output a list Z_x of strings such that $z_x \in Z_x$.

The algorithm M for deciding L by using the K(-) oracle is as follows. On input $x \in \{0,1\}^*$, enumerate the lists Y_x and Z_x. Accept if and only if there exists $y \in Y_x$ such that $V(x, y, z) = 1$ for every $z \in Z_x$. This algorithm runs in time $2^{O(\log^2 n)}$.

We claim the correctness of this algorithm. Suppose that $x \in L$. Then $V(x, y_x, z) = L(x) = 1$ for every z; thus M accepts. Now consider the case when $x \notin L$. We have $V(x, y, z_x) = L(x) = 0$ for every y; thus M rejects. □

8.3 Average-Case NP-Hardness of SAT-Oracle MINKT

In the case of resource-bounded Kolmogorov complexity, symmetry of information does not necessarily hold. It is, however, still possible to prove some hardness results without conditional Kolmogorov complexity. Specifically, we prove "DistNP-hardness of $\mathrm{MINKT}^{\mathsf{SAT}}$." Here $\mathrm{MINKT}^{\mathsf{SAT}}$ is an unconditional version of MINcKT, defined as:

$$\mathrm{MINKT}^{\mathsf{SAT}} := \{\, (x, 1^s, 1^t) \mid \mathrm{K}^{t,\mathsf{SAT}}(x) \leq s \,\}.$$

DistNP is the class of distributional problems $(L, \mathcal{D})$ such that $L \in \mathsf{NP}$ and $\mathcal{D}$ is efficiently samplable. (The reader is referred to the survey of Bogdanov and Trevisan [BT06] for more background on average-case complexity.) $\mathsf{HeurBPP}$ is defined as follows.

Definition 8.6 (Randomized Heuristic Scheme; cf. [BT06])**.** *We say that a distributional problem $(L, \mathcal{D})$ is in $\mathsf{HeurBPP}$ if there exists a randomized polynomial-time algorithm M such that, for all $n \in \mathbb{N}$,*

$$\Pr_{x,M}[M(x; 1^m) = L(x)] \geq 1 - \frac{1}{m},$$

where the probability is taken over a random choice of an input $x \sim \mathcal{D}_n$ as well as an internal randomness of M.

Restatement of Theorem 3.5. If $\mathrm{MINKT}^{\mathsf{SAT}} \in \mathsf{BPP}$, then $\mathsf{DistNP} \subseteq \mathsf{HeurBPP}$.

Proof. Let $\mathcal{U} = \{\mathcal{U}_n\}_{n\in\mathbb{N}}$ denote the family of the uniform distributions $\mathcal{U}_n$ on $\{0,1\}^n$. Impagliazzo and Levin [IL90] showed that $\mathsf{NP} \times \{\mathcal{U}\} \subseteq \mathsf{HeurBPP}$ implies that $\mathsf{DistNP} \subseteq \mathsf{HeurBPP}$; thus it suffices to prove a randomized heuristic scheme for an arbitrary language $L \in \mathsf{NP}$ under the uniform distribution.

Fix any input $(x, 1^m)$, where x is an input of length $n \in \mathbb{N}$ and m is a precision parameter of $\mathsf{HeurBPP}$. Let V be a polynomial-time verifier that witnesses $L \in \mathsf{NP}$. For each $x \in L$, we denote by y_x the lexicographically first certificate y_x such that $V(x, y_x) = 1$. Note that y_x is computable in P^{NP}; thus $\mathrm{K}^{t,\mathsf{SAT}}(y_x|x) = O(1)$ for a large polynomial t.

We will use the Randomized Enumeration Lemma (Lemma 5.5). To this end, we define an oracle D as

$$D(x, w) := 1 \quad \Longleftrightarrow \quad \mathrm{K}^{t,\mathsf{SAT}}(x, w) < |x| + |w| - \log m - 1,$$

for any $x, w \in \{0,1\}^*$, where t is some sufficiently large polynomial.

Consider the direct product generator $\mathrm{DP}_k^{\mathtt{fn}(y_x)} \colon \{0,1\}^d \to \{0,1\}^{d+k}$ of Definition 5.3, where $d = d(n, m)$ is the seed length of $\mathrm{DP}_k^{\mathtt{fn}(y_x)}$ and $k = k(n, m)$ is some parameter chosen later.

By using the counting argument of Fact 4.1, we obtain

$$\Pr_{\substack{x\sim\{0,1\}^n \\ w\sim\{0,1\}^{d+k}}} [D(x, w) = 1] \le \frac{1}{2m}.$$

Define $I := \{\, x \in \{0,1\}^* \mid \Pr_w [D(x, w) = 1] \le \frac{1}{2} \,\}$. Then, it follows from Markov's inequality that

$$\Pr_x [x \in I] \ge 1 - \frac{1}{m}. \tag{4}$$

We claim that, for every $x \in I$, the oracle $D(x, \text{-})$ distinguishes the output distribution of $\mathrm{DP}_{k(n)}^{\mathtt{fn}(y_x)}$ from the uniform distribution. Indeed, for any seed $z \in \{0,1\}^d$, the output $\mathrm{DP}_k^{\mathtt{fn}(y_x)}(z)$ can be described by a self-delimiting encoding of $n \in \mathbb{N}$, the input $x \in \{0,1\}^n$, and the seed $z \in \{0,1\}^d$; therefore,

$$\mathrm{K}^{t,\mathsf{SAT}}(x, \mathrm{DP}_k^{\mathtt{fn}(y_x)}(z)) \le n + d + O(\log n).$$

We choose a sufficiently large $k = \log m + O(\log n)$ so that this is less than $n + d + k - \log m - 1$. By the definition of D, we have $D(x, \mathrm{DP}_k^{\mathtt{fn}(y_x)}(z)) = 1$. Therefore, for any $x \in I$,

$$\Pr_z \left[D(x, \mathrm{DP}_k^{\mathtt{fn}(y_x)}(z)) = 1 \right] - \Pr_w [D(x, w) = 1] \ge 1 - \frac{1}{2} = \frac{1}{2}.$$

Applying Lemma 5.5 to the auxiliary-input function family $\{\mathtt{fn}(y_x)\}_{x\in I, m\in\mathbb{N}}$, we obtain a randomized D-oracle algorithm E, running in time $\mathsf{poly}(n, 2^k) = \mathsf{poly}(n, m)$, that, given $x \in I$ and $m \in \mathbb{N}$ as input, outputs a list of strings that contains y_x with high probability.

Under the assumption that $\mathrm{MINKT}^{\mathsf{SAT}} \in \mathsf{BPP}$, the oracle D can be computed in BPP. Replacing the D-oracle of E with the BPP algorithm, we obtain a randomized algorithm E' that enumerates a list Y of strings such that $y_x \in Y$, with high probability.

A randomized heuristic algorithm M for computing $(L, \mathcal{U})$ is as follows. On input $(x, 1^m)$, simulate E' to obtain a list Y of strings. Accept if and only if $V(x, y) = 1$ for some $y \in Y$.

We claim the correctness of M. It is clear that the randomized heuristic algorithm M rejects every input $x \notin L$. Now consider any input $x \in L$. By (4), with probability at least $1 - 1/m$ over

the choice of $x \sim \{0,1\}^n$, we have $x \in I$. Under this event, Lemma 5.5 guarantees that E' outputs a list Y such that $y_x \in Y$ with high probability (say, at least $1 - 1/m$) over the choice of internal randomness of E'. Therefore, with probability at least $1 - 1/2m$, the algorithm M accepts. □

9 Satisfiability Algorithms and Immunity of Random Strings

In this section, we forge a firm link between satisfiability algorithms for a uniform circuit class $\mathfrak{C}$ and $\mathfrak{C}$-immunity of resource-bounded Kolmogorov-random strings.

Definition 9.1 (Immunity)**.** *For a complexity class $\mathfrak{C}$, a language $L \subseteq \{0,1\}^*$ is called* $\mathfrak{C}$-immune *(or immune to $\mathfrak{C}$) if L is infinite and there exists no infinite subset $L' \subseteq L$ such that $L' \in \mathfrak{C}$.*

Theorem 9.2 (Non-trivial SAT $\Longrightarrow$ Immunity of R_{Kt})**.** *Let $\mathfrak{C}$ be an arbitrary circuit class. Suppose that $\mathfrak{C}$ admits a non-trivial satisfiability algorithm, i.e., there exists an algorithm C that, given an n-input circuit $C \in \mathfrak{C}$ as input, outputs an assignment $a \in \{0,1\}^n$ such that $C(a) = 1$ (if exists), and runs in time $|C|^{O(1)} \cdot 2^n/n^{\omega(1)}$. Then, the set R^s_{Kt} of s-random strings is immune to P-uniform $\mathfrak{C}$ for some time-constructible function $s(n) = n - \omega(\log n)$.*

Proof. Let M be the non-trivial satisfiability algorithm for $\mathfrak{C}$. Assume, towards a contradiction, that there exists an infinite subset $R \subseteq R_{\mathrm{Kt}}$ such that R is in P-uniform $\mathfrak{C}$. Let R_0 be a polynomial-time algorithm that, on input 1^n, prints a description of a $\mathfrak{C}$-circuit C_n that accepts $R \cap \{0,1\}^n$. Since R is infinite, there are infinitely many inputs $n \in \mathbb{N}$ such that C_n has at least one satisfying assignment. Fix such an input length $n \in \mathbb{N}$.

Now consider the output of the algorithm M on input C_n. Since C_n is satisfiable, M finds some assignment $a_n \in \{0,1\}^n$ such that $C_n(a_n) = 1$. By the assumption that $R \subseteq R_{\mathrm{Kt}}$ and C_n accepts $R \cap \{0,1\}^n$, we have $a_n \in R_{\mathrm{Kt}}$ and thus $\mathrm{Kt}(a_n) \geq s(n)$.

On the other hand, one can efficiently describe a_n in the following way: Given $n \in \mathbb{N}$, run R_0 on input 1^n to generate a circuit C_n. Run M on input C_n to obtain the assignment $a_n \in \{0,1\}^n$, and output a_n. This algorithm runs in time $2^{n-\omega(\log n)}$. Hence,

$$\mathrm{Kt}(a_n) < O(\log n) + \log(2^{n-\omega(\log n)}) = n - \omega(\log n) =: s(n),$$

which is a contradiction for a sufficiently large n. □

Next, we prove the converse direction of Theorem 9.2. We will show that $\mathfrak{C}$-immunity of random strings implies the existence of a non-trivial SAT algorithm for $\mathfrak{C}$. Since the notion of immunity is meaningful only for uniform circuit classes (cf. Remark 9.5), we formalize the notion of SAT algorithms for *uniform circuit classes*.

Definition 9.3 (Non-trivial SAT for P-uniform circuits)**.** *For a circuit class $\mathfrak{C}$, an algorithm A is said to be a* non-trivial satisfiability algorithm for P-uniform $\mathfrak{C}$ *if, for any family of P-uniform $\mathfrak{C}$-circuits $C := \{C_n\}_{n\in\mathbb{N}}$, for all sufficiently large $n \in \mathbb{N}$, given a description of C_n as input, A outputs an assignment $a \in \{0,1\}^n$ such that $C_n(a) = 1$ when C has a satisfying assignment.*

Theorem 9.4 (Immunity of $R_{\mathrm{Kt}} \Longrightarrow$ Non-trivial SAT)**.** *Let $s\colon \mathbb{N} \to \mathbb{N}$ be a time-constructible function such that $s(n) = n - \omega(\log n)$ for $n \in \mathbb{N}$. Let $\mathfrak{C}$ be any (uniform) circuit class. Suppose that the set R^s_{Kt} of s-random strings is $\mathfrak{C}$-immune. Then, there exists a non-trivial satisfiability algorithm for $\mathfrak{C}$.*

Proof. The idea is to make use of the strategy of Levin's universal search algorithm. Namely, we enumerate all non-s-random strings with respect to Kt in time $2^{s(n)+O(\log n)}$, and we argue that one of them is a satisfying assignment (if exists).

Consider the following algorithm A for solving the satisfiability of $\mathfrak{C}$. Given an n-input circuit $C \in \mathfrak{C}$ as input, for each string d of length at most $s(n)$, simulate the universal machine U on input d for $2^{s(n)-|d|}$ time steps, and let a denote the output of U if U halts. If $a \in \{0,1\}^n$ and $C(a) = 1$, then output a and halt. (That is, A enumerates all the strings $a \in \{0,1\}^n$ such that $\mathrm{Kt}(a) \leq s(n)$ and check whether a is a satisfying assignment.) The running time of A can be bounded above by $\sum_{k \leq s(n)} 2^k \cdot 2^{s(n)-k} \cdot |C|^{O(1)} \leq 2^{s(n)+O(\log n)} \leq 2^{n-\omega(\log n)}$.

We claim that the algorithm A is a non-trivial SAT algorithm for $\mathfrak{C}$. Take any family of circuits $C = \{C_n\}_{n \in \mathbb{N}} \in \mathfrak{C}$. Assume, towards a contradiction, that C_n has a satisfying assignment whereas A cannot find any satisfying assignment for C_n, for an infinitely many $n \in \mathbb{N}$. This means that the set $C_n^{-1}(1)$ of satisfying assignments is non-empty, and that $C_n^{-1}(1) \subseteq \{0,1\}^n$ is a subset of s-random strings R^s_{Kt}. However, this contradicts the assumption that R^s_{Kt} is $\mathfrak{C}$-immune. Thus A finds a satisfying assignment of C_n for all sufficiently large $n \in \mathbb{N}$. □

Remark 9.5. Theorem 9.4 holds for any (not necessarily uniform) circuit class $\mathfrak{C}$. However, for any reasonable non-uniform circuit class $\mathfrak{C}$, the hypothesis of Theorem 9.4 is always false, i.e., R_{Kt} is not $\mathfrak{C}$-immune. This is because of the fact that one random string can be provided as non-uniform advice. Therefore, Theorem 9.4 is meaningful only for uniform circuit classes.

These two directions can be used to show the equivalence between the existence of a non-trivial SAT algorithm for P-uniform $\mathfrak{C}$ and (P-uniform $\mathfrak{C}$)-immunity of R_{Kt}.

Proof of Theorem 3.11. We apply Theorem 9.2 and Theorem 9.4 to P-uniform $\mathfrak{C}$. Since P-uniform (P-uniform $\mathfrak{C}$) is equal to P-uniform $\mathfrak{C}$, we obtain the equivalence between non-trivial SAT and immunity for P-uniform $\mathfrak{C}$. □

Using the non-trivial SAT algorithm of Williams [Wil14], we will present a P-uniform ACC^0 circuit lower bound.

Lemma 9.6 (Williams [Wil14])**.** *For every $d, m > 1$, there exist $\delta, \epsilon > 0$ and an algorithm that decides the satisfiability of depth-d ACC^0 circuits with MOD_m gates, n inputs, and 2^{n^ϵ} size in time $2^{n-\Omega(n^\delta)}$.*

Reminder of Theorem 3.10. R_{Kt} is (P-uniform ACC^0)-immune.

Proof. The algorithm of Lemma 9.6 solves the decision version of SAT; by using the downward self-reducibility of SAT, the algorithm can be modified so that it solves the search version of SAT (with at most $\mathsf{poly}(n)$-factor overhead in the running time). Therefore, there exists a non-trivial satisfiability algorithm for ACC^0. The result follows from Theorem 9.2. □

Remark 9.7. By using the full power of Lemma 9.6, it is not hard to extend this result to the following: For every $d > 1, m > 1$, there exist $\delta, \epsilon > 0$ such that, R_{Kt} is immune to $\mathsf{DTIME}(2^{n^\delta})/n^\delta$-uniform depth-$d$ ACC^0 circuits with MOD_m gates of size 2^{n^ϵ}.

We have presented that a non-trivial satisfiability algorithm with respect to Levin's Kolmogorov complexity corresponds to the non-trivial satisfiability algorithm in the sense of [Wil14]. Surprisingly, in the case of K^t-complexity for a super-polynomial t, we can present a "*non-trivial satisfiability*" algorithm for P with respect to the K^t-complexity measure.

Reminder of Theorem 3.9. Let $t\colon \mathbb{N} \to \mathbb{N}$ be a function such that $t(n) = n^{\omega(1)}$ for $n \in \mathbb{N}$. Then, R_{K^t} is P-immune.

Proof. At a high level, the idea is to have an advice string a and then try to solve the satisfiability of $M(a,\text{-})$ by an exhaustive search for any polynomial-time algorithm M. We will show that this is a "non-trivial satisfiability" algorithm in some sense. A formal proof follows.

Assume, towards a contradiction, that $R \subseteq R_{\mathrm{K}^t}$ is an infinite subset in P, and that M is a polynomial-time algorithm that computes R.

Let $m\colon \mathbb{N} \to \mathbb{N}$ be some parameter chosen later (such that $m(n) = O(\log n)$). Consider the following algorithm A. The input of A consists of an "advice" string $a \in \{0,1\}^{n-m(n)}$. The algorithm A checks whether there exists a string $x \in \{0,1\}^{m(n)}$ such that $M(a,x) = 1$ by an exhaustive search, and outputs $(a,x) \in \{0,1\}^n$ and halts if such a string x is found. The running time of A is at most $2^{m(n)}n^{O(1)}$.

By the assumption, there exist infinitely many $n \in \mathbb{N}$ such that $R \cap \{0,1\}^n$ is not empty. Fix such an integer $n \in \mathbb{N}$. We take an arbitrary string $r_n \in R \cap \{0,1\}^n$, and let $a_n \in \{0,1\}^{n-m(n)}$ be the first $n - m(n)$ bits of r_n. Since the set $R'_n := \{\, (a_n,x) \in R \cap \{0,1\}^n \mid x \in \{0,1\}^{m(n)} \,\}$ is non-empty, the algorithm A finds some $x \in \{0,1\}^{m(n)}$ such that $M(a_n,x) = 1$, i.e., $(a_n,x) \in R \cap \{0,1\}^n$. The Kolmogorov complexity of the string $(a_n,x) \in \{0,1\}^n$ is at most

$$\mathrm{K}^{2^{m(n)}n^{O(1)}}(a_n,x) \leq n - m(n) + 2\log n + O(1), \tag{5}$$

because (a_n,x) can be described by using $n \in \mathbb{N}$, $a \in \{0,1\}^{n-m(n)}$, and A in time $2^{m(n)}n^{O(1)}$. We choose $m(n) = O(\log n)$ large enough so that the upper bound of (5) is less than $n-1$. For all sufficiently large $n \in \mathbb{N}$, we have $2^{m(n)}n^{O(1)} = n^{O(1)} \leq t(n)$, and thus

$$\mathrm{K}^t(a_n,x) < n - 1.$$

However, since $(a_n,x) \in R \subseteq R_{\mathrm{K}^t}$, we also have $\mathrm{K}^t(a_n,x) \geq n-1$, which is a contradiction. □

References

[ABFL14] Eric Allender, Harry Buhrman, Luke Friedman, and Bruno Loff. Reductions to the set of random strings: The resource-bounded case. *Logical Methods in Computer Science*, 10(3), 2014.

[ABK06a] Eric Allender, Harry Buhrman, and Michal Koucký. What can be efficiently reduced to the Kolmogorov-random strings? *Ann. Pure Appl. Logic*, 138(1-3):2–19, 2006.

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

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

[ADF+13] Eric Allender, George Davie, Luke Friedman, Samuel Hopkins, and Iddo Tzameret. Kolmogorov Complexity, Circuits, and the Strength of Formal Theories of Arithmetic. *Chicago J. Theor. Comput. Sci.*, 2013, 2013.

[AFG13] Eric Allender, Luke Friedman, and William I. Gasarch. Limits on the computational power of random strings. *Inf. Comput.*, 222:80–92, 2013.

[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.

[AHK17] Eric Allender, Dhiraj Holden, and Valentine Kabanets. The Minimum Oracle Circuit Size Problem. *Computational Complexity*, 26(2):469–496, 2017.

[AHM+08] Eric Allender, Lisa Hellerstein, Paul McCabe, Toniann Pitassi, and Michael E. Saks. Minimizing Disjunctive Normal Form Formulas and AC^0 Circuits Given a Truth Table. *SIAM J. Comput.*, 38(1):63–84, 2008.

[AKRR11] Eric Allender, Michal Koucký, Detlef Ronneburger, and Sambuddha Roy. The pervasive reach of resource-bounded Kolmogorov complexity in computational complexity theory. *J. Comput. Syst. Sci.*, 77(1):14–40, 2011.

[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.

[All17] Eric Allender. The Complexity of Complexity. In *Computability and Complexity - Essays Dedicated to Rodney G. Downey on the Occasion of His 60th Birthday*, pages 79–94, 2017.

[AS12] Eric Allender and Holger Spakowski. Avoiding Simplicity is Complex. *Theory Comput. Syst.*, 51(3):282–296, 2012.

[BFKL10] Harry Buhrman, Lance Fortnow, Michal Koucký, and Bruno Loff. Derandomizing from Random Strings. In *Proceedings of the Conference on Computational Complexity (CCC)*, pages 58–63, 2010.

[BFL91] László Babai, Lance Fortnow, and Carsten Lund. Non-Deterministic Exponential Time has Two-Prover Interactive Protocols. *Computational Complexity*, 1:3–40, 1991.

[BFLS91] László Babai, Lance Fortnow, Leonid A. Levin, and Mario Szegedy. Checking Computations in Polylogarithmic Time. In *Proceedings of the Symposium on Theory of Computing (STOC)*, pages 21–31, 1991.

[BFNW93] László Babai, Lance Fortnow, Noam Nisan, and Avi Wigderson. BPP Has Subexponential Time Simulations Unless EXPTIME has Publishable Proofs. *Computational Complexity*, 3:307–318, 1993.

[BGH+05] Eli Ben-Sasson, Oded Goldreich, Prahladh Harsha, Madhu Sudan, and Salil P. Vadhan. Short PCPs Verifiable in Polylogarithmic Time. In *Proceedings of the Conference on Computational Complexity (CCC)*, pages 120–134, 2005.

[BK95] Manuel Blum and Sampath Kannan. Designing Programs that Check Their Work. *J. ACM*, 42(1):269–291, 1995.

[BM97] Harry Buhrman and Elvira Mayordomo. An Excursion to the Kolmogorov Random Strings. *J. Comput. Syst. Sci.*, 54(3):393–399, 1997.

[BT00] Harry Buhrman and Leen Torenvliet. Randomness is Hard. *SIAM J. Comput.*, 30(5):1485–1501, 2000.

[BT06] Andrej Bogdanov and Luca Trevisan. On Worst-Case to Average-Case Reductions for NP Problems. *SIAM J. Comput.*, 36(4):1119–1159, 2006.

[Cai07] Jin-yi Cai. $\mathsf{S}_2^p \subseteq \mathsf{ZPP}^{\mathsf{NP}}$. *J. Comput. Syst. Sci.*, 73(1):25–35, 2007.

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

[CDE+14] Mingzhong Cai, Rodney G. Downey, Rachel Epstein, Steffen Lempp, and Joseph S. Miller. Random strings and tt-degrees of Turing complete C.E. sets. *Logical Methods in Computer Science*, 10(3), 2014.

[Coo71] Stephen A. Cook. The Complexity of Theorem-Proving Procedures. In *Proceedings of the Symposium on Theory of Computing (STOC)*, pages 151–158, 1971.

[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)*, pages 469–482, 2008.

[GW00] Oded Goldreich and Avi Wigderson. On Pseudorandomness with respect to Deterministic Observes. In *ICALP Satellite Workshops*, pages 77–84, 2000.

[Har83] Juris Hartmanis. Generalized Kolmogorov Complexity and the Structure of Feasible Computations (Preliminary Report). In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 439–445, 1983.

[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.

[Hir20] Shuichi Hirahara. Unexpected Power of Random Strings. In *Proceedings of the Innovations in Theoretical Computer Science Conference (ITCS)*, pages 41:1–41:13, 2020.

[HK18] Shuichi Hirahara and Akitoshi Kawamura. On characterizations of randomized computation using plain Kolmogorov complexity. *Computability*, 7(1):45–56, 2018.

[HOS18] Shuichi Hirahara, Igor Carboni Oliveira, and Rahul Santhanam. NP-hardness of Minimum Circuit Size Problem for OR-AND-MOD Circuits. In *Proceedings of the Computational Complexity Conference (CCC)*, pages 5:1–5:31, 2018.

[HP15] John M. Hitchcock and Aduri Pavan. On the NP-Completeness of the Minimum Circuit Size Problem. In *Proceedings of the Conference on Foundation of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 236–245, 2015.

[HS65] Juris Hartmanis and Richard E Stearns. On the computational complexity of algorithms. *Transactions of the American Mathematical Society*, 117:285–306, 1965.

[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.

[HW19] Shuichi Hirahara and Osamu Watanabe. On Nonadaptive Security Reductions of Hitting Set Generators. 2019. ECCC TR19-025.

[IKV18] Russell Impagliazzo, Valentine Kabanets, and Ilya Volkovich. The Power of Natural Properties as Oracles. In *Proceedings of the Computational Complexity Conference (CCC)*, pages 7:1–7:20, 2018.

[IL90] Russell Impagliazzo and Leonid A. Levin. No Better Ways to Generate Hard NP Instances than Picking Uniformly at Random. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 812–821, 1990.

[Ila20] Rahul Ilango. Approaching MCSP from Above and Below: Hardness for a Conditional Variant and $\mathsf{AC}^0[p]$. In *Proceedings of the Innovations in Theoretical Computer Science Conference (ITCS)*, pages 34:1–34:26, 2020.

[IW97] Russell Impagliazzo and Avi Wigderson. $P = BPP$ if E Requires Exponential Circuits: Derandomizing the XOR Lemma. In *Proceedings of the Symposium on the Theory of Computing (STOC)*, pages 220–229, 1997.

[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.

[Ko86] Ker-I Ko. On the Notion of Infinite Pseudorandom Sequences. *Theor. Comput. Sci.*, 48(3):9–33, 1986.

[Ko91] Ker-I Ko. On the Complexity of Learning Minimum Time-Bounded Turing Machines. *SIAM J. Comput.*, 20(5):962–986, 1991.

[Kre88] Mark W. Krentel. The Complexity of Optimization Problems. *J. Comput. Syst. Sci.*, 36(3):490–509, 1988.

[KS99] Ravi Kumar and D. Sivakumar. Proofs, Codes, and Polynomial-Time Reducibilities. In *Proceedings of the Conference on Computational Complexity (CCC)*, pages 46–53, 1999.

[Lev73] Leonid Anatolevich Levin. Universal sequential search problems. *Problemy Peredachi Informatsii*, 9(3):115–116, 1973.

[Lev84] Leonid A. Levin. Randomness Conservation Inequalities; Information and Independence in Mathematical Theories. *Information and Control*, 61(1):15–37, 1984.

[LR05] Troy Lee and Andrei E. Romashchenko. Resource bounded symmetry of information revisited. *Theor. Comput. Sci.*, 345(2-3):386–405, 2005.

[LV08] Ming Li and Paul M. B. Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications, Third Edition*. Texts in Computer Science. Springer, 2008.

[LW95] Luc Longpré and Osamu Watanabe. On Symmetry of Information and Polynomial Time Invertibility. *Inf. Comput.*, 121(1):14–22, 1995.

[Mas79] William J Masek. Some NP-complete set covering problems. *Unpublished manuscript*, 1979.

[MW17] Cody D. Murray and R. Ryan Williams. On the (Non) NP-Hardness of Computing Circuit Complexity. *Theory of Computing*, 13(1):1–22, 2017.

[Oli19] Igor Carboni Oliveira. Randomness and Intractability in Kolmogorov Complexity. In *Proceedings of the International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 32:1–32:14, 2019.

[Ron04] Detlef Ronneburger. Kolmogorov Complexity and Derandomization. 2004.

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

[Sip83] Michael Sipser. A Complexity Theoretic Approach to Randomness. In *Proceedings of the Symposium on Theory of Computing (STOC)*, pages 330–335, 1983.

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

[Sud97] Madhu Sudan. Decoding of Reed Solomon Codes beyond the Error-Correction Bound. *J. Complexity*, 13(1):180–193, 1997.

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

[Vad12] Salil P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1-3):1–336, 2012.

[VV83] Umesh V. Vazirani and Vijay V. Vazirani. A Natural Encoding Scheme Proved Probabilistic Polynomial Complete. *Theor. Comput. Sci.*, 24:291–300, 1983.

[Wil83] Robert E. Wilber. Randomness and the Density of Hard Problems. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 335–342, 1983.

[Wil13] Ryan Williams. Improving Exhaustive Search Implies Superpolynomial Lower Bounds. *SIAM J. Comput.*, 42(3):1218–1244, 2013.

[Wil14] Ryan Williams. Nonuniform ACC Circuit Lower Bounds. *J. ACM*, 61(1):2:1–2:32, 2014.

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il