Electronic Colloquium on Computational Complexity, Report No. 143 (2020)

# Characterizing Average-Case Complexity of PH by Worst-Case Meta-Complexity

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

September 16, 2020

**Abstract**

We exactly characterize the average-case complexity of the polynomial-time hierarchy ($\mathsf{PH}$) by the worst-case (meta-)complexity of $\mathrm{GapMINKT}^{\mathsf{PH}}$, i.e., an approximation version of the problem of determining if a given string can be compressed to a short $\mathsf{PH}$-oracle efficient program. Specifically, we establish the following equivalence:

$$\mathsf{DistPH} \subseteq \mathsf{AvgP} \ (\text{i.e., } \mathsf{PH} \text{ is easy on average}) \iff \mathrm{GapMINKT}^{\mathsf{PH}} \in \mathsf{P}.$$

In fact, our equivalence is significantly broad: A number of statements on several fundamental notions of complexity theory, such as errorless and one-sided-error average-case complexity, sublinear-time-bounded and polynomial-time-bounded Kolmogorov complexity, and $\mathsf{PH}$-computable hitting set generators, are all shown to be equivalent.

Our equivalence provides fundamentally new proof techniques for analyzing average-case complexity through the lens of *meta-complexity* of time-bounded Kolmogorov complexity and resolves, as immediate corollaries, questions of equivalence among different notions of average-case complexity of $\mathsf{PH}$: low success versus high success probabilities (i.e., a hardness amplification theorem for $\mathsf{DistPH}$ against uniform algorithms) and errorless versus one-sided-error average-case complexity of $\mathsf{PH}$.

Our results are based on a sequence of new technical results that further develops the proof techniques of the author's previous work on the non-black-box worst-case to average-case reduction and unexpected hardness results for Kolmogorov complexity (FOCS'18, CCC'20, ITCS'20, STOC'20). Among other things, we prove the following.

1. $\mathrm{GapMINKT}^{\mathsf{NP}} \in \mathsf{P}$ implies $\mathsf{P} = \mathsf{BPP}$. At the core of the proof is a new black-box hitting set generator construction whose reconstruction algorithm uses few random bits, which also improves the approximation quality of the non-black-box worst-case to average-case reduction without using a pseudorandom generator.
2. $\mathrm{GapMINKT}^{\mathsf{PH}} \in \mathsf{P}$ implies $\mathsf{DistPH} \subseteq \mathsf{AvgBPP} = \mathsf{AvgP}$.
3. If $\mathrm{MINKT}^{\mathsf{PH}}$ is easy on a $1/\mathsf{poly}(n)$-fraction of inputs, then $\mathrm{GapMINKT}^{\mathsf{PH}} \in \mathsf{P}$. This improves the error tolerance of the previous non-black-box worst-case to average-case reduction.



ISSN 1433-8092

# Contents

# 1 Introduction

Two of the mysteries of complexity theory are the *average-case complexity* of $\mathsf{PH}$ and the computational complexity of computing time-bounded Kolmogorov complexity (referred to as *meta-complexity*). These questions originate from the 1980s, when Levin [Lev86] laid the foundation of the average-case complexity theory and Ko [Ko91] investigated the complexity of MINKT, which is the problem of computing time-bounded Kolmogorov complexity. The exact relationship among them has not been well understood. In this paper, we establish an interdisciplinary link between the two subareas of complexity theory.

**Main Theorem** (a short version; "$\mathsf{DistPH}$-completeness" of $\mathrm{GapMINKT}^{\mathsf{PH}}$)**.**

$$\mathsf{DistPH} \subseteq \mathsf{AvgP} \iff \mathrm{GapMINKT}^{\mathsf{PH}} \in \mathsf{P}.$$

This equivalence connects two fundamentally different notions. On the left, the statement $\mathsf{DistPH} \subseteq \mathsf{AvgP}$ means that $\mathsf{PH}$ is easy on *most* instances. On the right, the statement $\mathrm{GapMINKT}^{\mathsf{PH}} \in \mathsf{P}$ means that an efficient algorithm can compute the $\mathsf{PH}$-oracle Kolmogorov complexity of *every* instance. In fact, our equivalence is much larger, and it connects a number of statements on several notions of complexity theory, including errorless and one-sided-error average-case complexity, time-bounded Kolmogorov complexity, and $\mathsf{PH}$-computable hitting set generator.

Our equivalence not only is interdisciplinary but also has significant impacts on fundamental questions in each subarea. We review the average-case complexity theory and its open questions in the next section, as well as the notion of time-bounded Kolmogorov complexity in the subsequent section.

## 1.1 Average-Case Complexity Theory

In practice, the traditional analysis of an algorithm based on *worst-case* inputs can be misleading. It is often reported that modern $\mathsf{SAT}$ solvers can solve huge instances, notwithstanding the $\mathsf{NP}$-completeness of $\mathsf{SAT}$. The main source of the disagreement between the practical performance of an algorithm and the $\mathsf{NP}$-completeness theory is that the latter notion is based on the *worst-case* analysis of an algorithm; however, we cannot generate worst-case inputs, and thus never encounter them in reality.

Pioneered by Levin [Lev86], the theory of *average-case complexity* aims at analyzing the performance of algorithms with respect to random inputs sampled efficiently from some distribution. Specifically, for a language $L\colon \{0,1\}^* \to \{0,1\}$ and a family of distributions $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$,[1] the pair $(L,\mathcal{D})$ is called a *distributional problem*. The task of the distributional problem $(L,\mathcal{D})$ is to compute $L(x)$ given a random input $x \sim \mathcal{D}_n$ for each instance size $n \in \mathbb{N}$.[2] Following Levin's original notion [Lev86], $(L,\mathcal{D})$ is said to be *polynomial-time-solvable on average* if there exists an algorithm $A$ computing $L$ such that, for some constant $\epsilon > 0$, for all large $n \in \mathbb{N}$, $\mathbb{E}_{x\sim\mathcal{D}_n}[t_A(x)^\epsilon] \le O(n)$, where $t_A(x)$ denotes the running time of $A$ on input $x$. The class of distributional problems that are polynomial-time-solvable on average is denoted by $\mathsf{AvgP}$.

For our purpose, it is useful to use the equivalent notion of an *errorless heuristic scheme*. An errorless heuristic scheme $A$ for a distributional problem $(L,\mathcal{D})$ satisfies the following: (1) $A$ takes an input $x \in \mathsf{supp}(\mathcal{D}_n) \subseteq \{0,1\}^*$ and an error parameter $\delta \in (0,1)$ and runs in time $\mathsf{poly}(n, 1/\delta)$, (2)

[1] We identify a language $L \subseteq \{0,1\}^*$ with its characteristic function $L\colon \{0,1\}^* \to \{0,1\}$.

[2] The support of $\mathcal{D}_n$ is a subset of $\{0,1\}^*$, and it is not required to be a subset of $\{0,1\}^n$.

$A$ is errorless, that is, $A(x,\delta) \in \{L(x), \perp\}$, and (3) $A(x,\delta) = L(x)$ holds with probability at least $1-\delta$ over the choice of $x \sim \mathcal{D}_n$ for any $n \in \mathbb{N}$. It is known that $(L,\mathcal{D}) \in \mathsf{AvgP}$ if and only if $(L,\mathcal{D})$ admits an errorless heuristic scheme. (Roughly speaking, an errorless heuristic scheme outputs $\perp$ if it takes super-polynomial time to compute.) For a function $\delta\colon \mathbb{N} \to (0,1)$, let $\mathsf{Avg}_\delta\mathsf{P}$ denote the class of distributional problems $(L,\mathcal{D})$ that admit an errorless heuristic algorithm with error parameter fixed to $\delta(n)$. Further details on average-case complexity can be found in the survey of Bogdanov and Trevisan [BT06a].

What kind of distributions should we consider? It is reasonable to focus on the distribution from which one can generate a random instance efficiently. A family of distributions $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$ is said to be *polynomial-time samplable* if there exists a randomized polynomial-time algorithm that, on input $1^n$, outputs a string that is distributed according to $\mathcal{D}_n$. The class of polynomial-time-samplable distributions is denoted by $\mathsf{PSamp}$. For a complexity class $\mathfrak{C}$, let $\mathsf{Dist}\mathfrak{C}$ denotes the class $\mathfrak{C} \times \mathsf{PSamp}$ of distributional problems.

The fundamental questions of average-case complexity are whether $\mathsf{DistNP} \subseteq \mathsf{AvgP}$ and its relationship with $\mathsf{DistNP} \subseteq \mathsf{Avg}_\delta\mathsf{P}$ holds for different choices of parameters $\delta$. It is evident that $\mathsf{DistNP} \subseteq \mathsf{Avg}_\delta\mathsf{P}$ implies $\mathsf{DistNP} \subseteq \mathsf{Avg}_{\delta'}\mathsf{P}$ if $\delta \le \delta'$. However, it is a fundamental open question to prove the converse, depending on the choice of the error parameter $\delta$. For example, in the extreme case of $\delta(n) = 2^{-n}$, an errorless heuristic algorithm is equivalent to a worst-case solver, whose relationship with an errorless heuristic scheme is open.

**Open Question 1.1** (Worst-case versus average-case complexity of $\mathsf{NP}$)**.**

$$\text{Does } \mathsf{DistNP} \subseteq \mathsf{AvgP} \text{ imply } \mathsf{DistNP} \subseteq \mathsf{Avg}_{2^{-n}}\mathsf{P} \ (\Leftrightarrow \mathsf{NP} = \mathsf{P})?$$

This is one of the central questions in complexity theory, particularly because of its relationship to cryptography. In terms of Impagliazzo's five possible worlds, the open question corresponds to excluding Heuristica from the possible worlds (cf. [Imp95a]). In another regime of parameters, the question is referred to as *hardness amplification*.

**Open Question 1.2** (Hardness amplification for $\mathsf{NP}$)**.**

$$\text{Does } \mathsf{DistNP} \subseteq \mathsf{Avg}_{1-1/\mathsf{poly}(n)}\mathsf{P} \text{ imply } \mathsf{DistNP} \subseteq \mathsf{AvgP}?$$

The history of hardness amplification dates back to Yao's XOR Lemma (cf. [GNW11]); however, it is relatively recently that Bogdanov and Safra [BS07] initiated the study of hardness amplification in the context of *errorless* average-case complexity, and made progress towards Open Question 1.2 by showing that $\mathsf{DistNP} \subseteq \mathsf{Avg}_{1-(\log n)^{-1/10+o(1)}}\mathsf{P}$ implies $\mathsf{DistNP} \subseteq \mathsf{AvgP}$.[3]

Another natural question is whether two-sided-error average-case complexity of $\mathsf{NP}$ is equivalent to errorless average-case complexity, as raised in [Imp95a].

**Open Question 1.3** (Two-sided-error versus errorless average-case complexity of $\mathsf{NP}$)**.**

$$\text{Does } \mathsf{DistNP} \subseteq \mathsf{HeurP} \text{ imply } \mathsf{DistNP} \subseteq \mathsf{AvgP}?$$

Here, $\mathsf{HeurP}$ denotes the class of distributional problems that admit a two-sided-error heuristic scheme. In fact, using the search-to-decision reduction for $\mathsf{NP}$ [BCGL92], a two-sided-error heuristic scheme for $\mathsf{NP}$ can be converted to a "one-sided-error" heuristic scheme for $\mathsf{NP}$ that always rejects No instances and accepts most Yes instances. In light of this, Open Question 1.3 is morally equivalent to the following question.

[3] While their reduction is randomized, the reduction can be derandomized by using the pseudorandom generator of [BFP05].

**Open Question 1.4** (One-sided-error versus errorless average-case complexity of NP)**.**

$$\text{Does } \mathsf{DistNP} \subseteq \mathsf{Avg}^1\mathsf{P} \text{ imply } \mathsf{DistNP} \subseteq \mathsf{AvgP}?$$

Here, $\mathsf{Avg}^1\mathsf{P}$ denotes the class of distributional problems that admit a one-sided-error heuristic scheme. Specifically, we say that $(L, \mathcal{D}) \in \mathsf{Avg}^1\mathsf{P}$ if there exists an algorithm $A$ such that (1) $A(x, \delta)$ runs in time $\mathsf{poly}(n, 1/\delta)$, (2) $L(x) = 0$ implies $A(x, \delta) = 0$, and (3) $A(x, \delta) = L(x)$ with probability at least $1 - \delta$ over the choice of $x \sim \mathcal{D}_n$. (See Definition 3.2 for the precise definition of $\mathsf{Avg}^1\mathsf{P}$.)

So far we have explained the case of DistNP. However, all corresponding questions are open even for the Polynomial-time Hierarchy (PH). Recall that PH is a generalization of NP and is defined as $\bigcup_{k\in\mathbb{N}} \Sigma_k^{\mathrm{p}}$, where the $k$-th level $\Sigma_k^{\mathrm{p}}$ of PH is defined as $\mathsf{NP}^{\Sigma_{k-1}^{\mathrm{p}}}$ and $\Sigma_0^{\mathrm{p}} := \mathsf{P}$. For example, the following is an easier question than Open Question 1.1.

**Open Question 1.5** (Worst-case versus average-case complexity of PH)**.**

$$\text{Does } \mathsf{DistPH} \subseteq \mathsf{AvgP} \text{ imply } \mathsf{PH} = \mathsf{P} \ (\Leftrightarrow \mathsf{NP} = \mathsf{P})?$$

The landscape of average-case complexity is summarized in Fig. 1. The depicted implications are trivial facts, and the converse directions correspond to the open questions mentioned above.

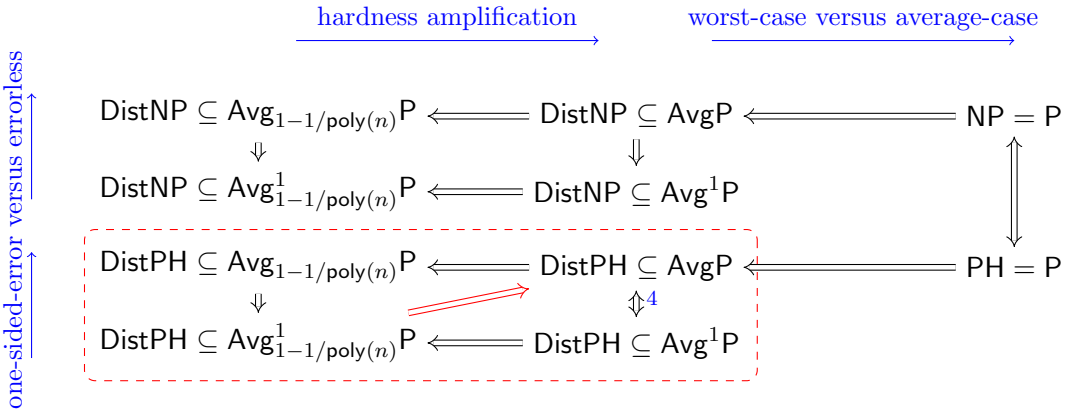


Figure 1: The relationships between average-case complexity of NP and PH under various notions. All implications depicted (except for our results) are trivial. The converse directions are fundamental open questions. A section of our equivalence is the four statements enclosed by the rectangle, which resolves the open questions regarding the average-case complexity of PH.

As a part of our equivalence, we resolve the PH analogues of Open Questions 1.2 and 1.4 simultaneously.[4] Specifically, we show that, if there exists a *one-sided-error* algorithm for DistPH that succeeds with probability at least $1/\mathsf{poly}(n)$, then there exists an *errorless* heuristic scheme for DistPH.

**Theorem 1.6.** *For any constant $c > 0$, if $\mathsf{DistPH} \subseteq \mathsf{Avg}^1_{1-n^{-c}}\mathsf{P}$, then $\mathsf{DistPH} \subseteq \mathsf{AvgP}$.*

[4]By using the fact that PH is closed under complement, we present in Appendix A a simple argument showing that $\mathsf{DistPH} \subseteq \mathsf{Avg}^1\mathsf{P}$ iff $\mathsf{DistPH} \subseteq \mathsf{AvgP}$. However, the same argument does not work when the failure probability is large.

Our proof techniques are fundamentally different from previous techniques. All previous proof techniques of hardness amplification that we are aware of use a hardness amplification procedure "Amp($f$)" (e.g., [Lev87, Imp95b, IW97, GNW11, O'D04, HVV06, Tre05, BS07, IJK09, IJKW10]). Specifically, a given function $f$ is mapped to a candidate hard function Amp($f$); typically, the function is defined as $\mathrm{Amp}(f)(x_1, \ldots, x_k) := f(x_1) \oplus \cdots \oplus f(x_k)$ as in Yao's XOR Lemma. Then, one designs an efficient oracle algorithm $R$ that, given access to an oracle that solves Amp($f$) on a small fraction of inputs, solves $f$ on a large fraction of inputs.

The proof strategy based on Amp($f$) is often referred to as *black-box*: The oracle algorithm $R$ takes as oracle an *arbitrary* (even inefficient) heuristic algorithm for Amp($f$) and solves $f$ on average. Such a proof technique based on a black-box hardness amplification procedure is so general that there are a number of significant limits (e.g., [Vio05, LTW08, SV10, GSV18, FF93, BT06b]). For example, Viola [Vio05] showed that the worst-case-to-average-case equivalence of PH (i.e., Open Question 1.5) cannot be proved by using a black-box hardness amplification procedure.

Note that, in order to prove a hardness amplification theorem, it suffices to design a "*non-black-box*" oracle algorithm $R$ that is successful only when the oracle is efficient. Here, a reduction is referred to as *non-black-box* if the proof of the correctness of the reduction uses the efficiency of the oracle in an essential manner.

The approach of this work is not black-box,[5] and our proof is based on new understanding of average-case complexity through the *meta-complexity* of time-bounded Kolmogorov complexity. In order to connect average-case complexity theory and meta-complexity theory, we employ the proof techniques of the author's previous work on a non-black-box worst-case-to-average-case reduction [Hir18], which overcame another fundamental limit of black-box reductions that was presented by Feigenbaum and Fortnow [FF93] and Bogdanov and Trevisan [BT06b]. We next review the notion of meta-complexity.

## 1.2 Meta-Complexity of Time-Bounded Kolmogorov Complexity

Kolmogorov complexity enables quantifying how much a string is complex. Informally, the *$t$-time-bounded Kolmogorov complexity* of a finite string $x \in \{0,1\}^*$ is defined as the minimum size of a program that outputs $x$ in $t$ steps. For a formal definition, we need to clarify the meaning of "the size of a program" by fixing a particular interpreter. We fix an efficient universal Turing machine $U$. The $t$-time-bounded Kolmogorov complexity $\mathrm{K}^t(x)$ of $x$ is formally defined as follows. (We adopt the definition that is meaningful even for $t \leq |x|$.)

**Definition 1.7.** *For a string $x \in \{0,1\}^*$, an oracle $A \subseteq \{0,1\}^*$, and a time bound $t \in \mathbb{N} \cup \{\infty\}$,*

$$\mathrm{K}^{t,A}(x) := \min\{\, |d| \mid U^{d,A}(i) \text{ outputs } x_i \text{ in time } t \text{ for each } i \in [|x|+1] \,\}.$$

*Here, $U^{d,A}(i)$ means the output of the universal Turing machine given random access to $d$ and $A$ and $i$ as input; $x_i$ denotes the $i$th bit of $x$ if $i \leq |x|$ and $\perp$ otherwise. We omit the superscript $A$ if $A = \varnothing$, and the superscript $t$ if $t = \infty$, respectively.*

Note that time-bounded Kolmogorov complexity itself asks the *complexity* of a shortest program to print a given string. Stepping back, one can ask the *complexity* of computing time-bounded Kolmogorov complexity—what is called *meta-complexity* of time-bounded Kolmogorov complexity.

[5] Although the proof is non-black-box, we do not know whether our hardness amplification result itself (Theorem 1.6) is subject to some black-box barrier results, such as [Vio05, LTW08]. See also Section 1.4.4.

Although the origin of meta-complexity can be traced back to the Russian study of 1950s [Tra84], its importance was identified only recently, especially through the study of the Minimum Circuit Size Problem (MCSP [KC00]).[6]

Among the early studies on meta-complexity of time-bounded Kolmogorov complexity, Ko [Ko91] introduced and investigated MINKT, which is the problem of deciding, given $(x, 1^t, 1^s)$ as input, whether there is a program of size at most $s$ that prints $x$ in time $t$, as well as its approximation version which we denote by GapMINKT. The problem GapMINKT asks for approximating $\mathrm{K}^t(x)$ within an additive error of $O(\log(|x| + t))$. Formally:

**Definition 1.8.** *For a function* $\tau\colon \mathbb{N}\times\mathbb{N} \to \mathbb{N}$ *and an oracle* $A \subseteq \{0,1\}^*$, $\mathrm{Gap}_\tau\mathrm{MINKT}^A$ *is defined as the promise problem* $(\Pi_{\mathrm{YES}}, \Pi_{\mathrm{NO}})$ *such that*

$$\Pi_{\mathrm{YES}} := \{\, (x, 1^t, 1^s) \mid \mathrm{K}^{t,A}(x) \le s \,\},$$
$$\Pi_{\mathrm{NO}} := \{\, (x, 1^t, 1^s) \mid \mathrm{K}^{\tau(|x|,t),A}(x) > s + \log \tau(|x|, t) \,\}.$$

*We say* $\mathrm{GapMINKT}^A \in \mathsf{P}$ *if there exists some polynomial* $\tau$ *such that* $\mathrm{Gap}_\tau\mathrm{MINKT}^A \in \mathsf{P}$. *We define* $\mathrm{MINKT}^A := (\Pi_{\mathrm{YES}}, \{0,1\}^* \setminus \Pi_{\mathrm{YES}})$.

We extend this definition to $\mathrm{GapMINKT}^{\mathfrak{C}}$ for a complexity class $\mathfrak{C}$. We could have simply defined $\mathrm{GapMINKT}^{\mathsf{PH}}$ as $\mathrm{GapMINKT}^A$ if we had a $\mathsf{PH}$-complete problem $A$. However, this definition is problematic: If there exists a $\mathsf{PH}$-complete problem $A$, then we must have $\mathsf{PH} \le^p_{\mathrm{m}} A \in \Sigma^{\mathrm{p}}_k$ for some constant $k \in \mathbb{N}$, which implies the unlikely consequence that the polynomial-time hierarchy collapses. We address this issue by introducing the definition of $\mathrm{GapMINKT}^{\mathfrak{C}}$ that does not depend on a complete problem, where $\mathfrak{C}$ is an arbitrary complexity class.

**Definition 1.9.** *For a complexity class* $\mathfrak{C}$, *we regard* $\mathrm{GapMINKT}^{\mathfrak{C}}$ *as a family of problems* $\{\, \mathrm{GapMINKT}^A \mid A \in \mathfrak{C} \,\}$. *We say* $\mathrm{GapMINKT}^{\mathfrak{C}} \in \mathsf{P}$ *if* $\mathrm{GapMINKT}^A \in \mathsf{P}$ *for any oracle* $A \in \mathfrak{C}$. *Similarly, we define* $\mathrm{MINKT}^{\mathfrak{C}} := \{\, \mathrm{MINKT}^A \mid A \in \mathfrak{C} \,\}$.

One of the central questions on GapMINKT is to classify its complexity. It is easy to observe that $\mathrm{GapMINKT} \in \mathsf{NP}$, and more generally, $\mathrm{GapMINKT}^A \in \mathsf{NP}^A$ for any oracle $A$. Thus, the central question is to prove its $\mathsf{NP}$-hardness, which would completely classify the complexity of GapMINKT as an "$\mathsf{NP}$-complete" problem.[7]

**Open Question 1.10.** Does $\mathrm{GapMINKT} \in \mathsf{P}$ imply $\mathsf{P} = \mathsf{NP}$?

Somewhat surprisingly, the following easier question is open as well.

**Open Question 1.11** ("$\mathsf{NP}$-hardness" of $\mathrm{GapMINKT}^{\mathsf{PH}}$)**.** Does $\mathrm{GapMINKT}^{\mathsf{PH}} \in \mathsf{P}$ imply $\mathsf{P} = \mathsf{NP}$?

Readers unfamiliar with meta-complexity may wonder why it is not trivial that $\mathsf{NP}$ is reducible to $\mathrm{GapMINKT}^{\mathsf{NP}}$, or more generally, that $A$ is reducible to $\mathrm{GapMINKT}^A$ for any oracle $A$. Intuitively, $\mathrm{GapMINKT}^A$ seems to be more difficult than computing $A$. Unfortunately, there is a gap between this intuition and actually constructing a reduction from $A$ to $\mathrm{GapMINKT}^A$: The meta-complexity

[6] The reader is referred to the survey of Allender [All17] for more details on meta-complexity and MCSP.

[7] We often use the weak notion of hardness and completeness. For example, we say that a problem $L$ is "$\mathsf{NP}$-complete" if $L \in \mathsf{P}$ implies $\mathsf{P} = \mathsf{NP}$ and $L \in \mathsf{NP}$. This property is implied by the standard $\mathsf{NP}$-completeness under polynomial-time reductions, but the converse is not necessarily true.

of $\mathrm{GapMINKT}^A$ refers to the complexity of minimizing the size of an $A$-oracle program; it is not clear whether the task of computing $A$ can be converted to the task of minimizing the size of an $A$-oracle program. In fact, this gap—which seems to be intuitively small—is the only missing piece for establishing the equivalence between worst-case and average-case complexity of $\mathsf{PH}$.

**Corollary 1.12** (of Main Theorem)**.** *Open Question 1.11 is* equivalent *to Open Question 1.5. That is, "*$\mathsf{NP}$*-hardness" of* $\mathrm{GapMINKT}^{\mathsf{PH}}$ *is equivalent to establishing the equivalence between worst-case and average-case complexity of* $\mathsf{PH}$*.*

One of our main technical contributions is to partially close the gap between the tasks of computing $A$ and minimizing an $A$-oracle program: We will show that $\mathrm{GapMINKT}^{\mathsf{PH}} \in \mathsf{P}$ implies $\mathsf{DistPH} \subseteq \mathsf{AvgP}$, which can be regarded as "$\mathsf{DistPH}$-hardness" of $\mathrm{GapMINKT}^{\mathsf{PH}}$. In fact, our results classify the complexity of $\mathrm{GapMINKT}^{\mathsf{PH}}$ as a "$\mathsf{DistPH}$-complete" problem in the following sense: $\mathsf{DistPH} \subseteq \mathsf{AvgP} \iff \mathrm{GapMINKT}^{\mathsf{PH}} \in \mathsf{P}$, thereby resolving the fundamental open question of the complexity of $\mathrm{GapMINKT}^{\mathsf{PH}}$.

**Monotonicity.** There is another counterintuitive property of meta-complexity that is not well understood—the *monotonicity* of meta-complexity. One might think that it is trivial that $\mathrm{GapMINKT}^{\mathsf{SAT}} \in \mathsf{P}$ implies $\mathrm{GapMINKT} \in \mathsf{P}$. However, this was not known before this work.[8]

More generally, one might guess that GapMINKT should be reducible to $\mathrm{GapMINKT}^A$ for any oracle $A$ via, for instance, the identity reduction that maps an instance to itself. While the identity reduction can map any YES instance of GapMINKT to a YES instance of $\mathrm{GapMINKT}^A$, it does not necessarily map a NO instance of GapMINKT to a NO instance of $\mathrm{GapMINKT}^A$. The identity reduction actually works under the notion of the average-case complexity of GapMINKT [HS17], but, in terms of worst-case complexity, the reduction may not be correct. In fact, any deterministic reduction does not work: there exists some oracle $A$ such that MCSP is not reducible to $\mathrm{MCSP}^A$ via any deterministic polynomial-time Turing reduction unless $\mathrm{MCSP} \in \mathsf{P}$ [HW16].

Nevertheless, we establish the following monotonicity property of meta-complexity.

**Theorem 1.13.** *Let $A$ and $B$ be oracles such that $A$ is* $\mathsf{NP}$*-hard and $B \leq_{\mathrm{T}}^{p} A$. Then,* $\mathrm{GapMINKT}^A \in \mathsf{P}$ *implies* $\mathrm{GapMINKT}^B \in \mathsf{P}$

The proof of Theorem 1.13 is based on non-black-box reductions, thereby bypassing the impossibility result of [HW16]. At the core of the proof is the randomized non-black-box worst-case-to-average-case reduction of [Hir18, Hir20a]. In order to derandomize it, we again use a (deterministic) non-black-box worst-case-to-average-case reduction to construct a nearly optimal pseudorandom generator, by making use of the $\mathsf{NP}$-hardness of the oracle $A$. The details can be found in Section 8.2.

## 1.3 Our Results

Thus far, we have explained the significance of our results mainly within each subarea of complexity theory. We now guide the reader to our interdisciplinary equivalence that connects average-case complexity, meta-complexity of time-bounded Kolmogorov complexity, and more. Since the number of equivalent statements is large, we will explain one by one while presenting some ideas of the proofs.

[8] In contrast, it is easy to observe that $\mathrm{GapMINKT}^{\mathsf{NP}} \in \mathsf{P}$ implies $\mathrm{GapMINKT} \in \mathsf{P}$.

**Theorem 1.14** (Main results)**.** *The following (Items 1 to 12) are equivalent.*

1. $\mathsf{DistPH} \subseteq \mathsf{AvgP}$.

2. $\mathrm{GapMINKT}^{\mathsf{PH}} \in \mathsf{P}$.

Our equivalence is considerably robust with respect to minor changes. For example, recall that $\mathrm{GapMINKT}^{\mathsf{PH}}$ is the family of problems $\mathrm{GapMINKT}^A$ for each oracle $A \in \mathsf{PH}$. Although this is a convenient notion, we do not have to consider every oracle $A \in \mathsf{PH}$; alternatively, it suffices to consider a complete problem for each level of $\mathsf{PH}$. Let $\Sigma_k\mathsf{SAT}$ denote the canonical complete problem for $\Sigma_k^{\mathrm{p}}$. Then the following is equivalent as well.

3. $\mathrm{GapMINKT}^{\Sigma_k\mathsf{SAT}} \in \mathsf{P}$ *for any constant* $k \in \mathbb{N}$.

The significance of our equivalence is that the robustness of meta-complexity can be transferred to average-case complexity, the latter of which is often not resilient to modifications of success probability or the notion of errorless to one-sided-error. The equivalence enables us to reduce the success probability of an errorless heuristic algorithm to $1/\mathsf{poly}(n)$, which establishes a hardness amplification theorem for $\mathsf{PH}$ against uniform algorithms.

4. $\mathsf{DistPH} \subseteq \mathsf{Avg}_{1-n^{-c}}\mathsf{P}$ *for some constant* $c > 0$.

Furthermore, the equivalence extends to one-sided-error heuristic algorithms, thereby equating the errorless and one-sided-error average-case complexity of $\mathsf{PH}$. We are unaware of any existing proof techniques that can yield such an equivalence.[9]

5. $\mathsf{DistPH} \subseteq \mathsf{Avg}^1_{1-n^{-c}}\mathsf{P}$ *for some constant* $c > 0$.

How do we establish the equivalence? Essential to our proof is to identify $\mathrm{MINKT}^{\mathsf{PH}} \times \mathsf{PSamp}$ as a natural "$\mathsf{DistPH}$-complete" family of distributional problems.[10] Here, we say that a class $\mathfrak{C}$ of distributional problems is "$\mathsf{DistPH}$-complete" if $\mathfrak{C} \subseteq \mathsf{DistPH}$, and $\mathfrak{C} \subseteq \mathsf{AvgP}$ implies $\mathsf{DistPH} \subseteq \mathsf{AvgP}$. We show that $\mathrm{MINKT}^{\mathsf{PH}} \times \mathsf{PSamp}$ is "$\mathsf{DistPH}$-hard" even if the success probability of an errorless heuristic algorithm is $1/\mathsf{poly}(n)$.

6. $\mathsf{Dist}(\mathrm{MINKT}^{\mathsf{PH}}) := \mathrm{MINKT}^{\mathsf{PH}} \times \mathsf{PSamp} \subseteq \mathsf{Avg}_{1-n^{-c}}\mathsf{P}$ *for some constant* $c > 0$.

The reason why we are able to show the equivalence between one-sided-error and errorless average-case complexity is that $\mathsf{Dist}(\mathrm{MINKT}^{\mathsf{PH}})$ is "$\mathsf{DistPH}$-hard" in an even stronger sense. If $\mathsf{coMINKT}^{\mathsf{PH}}$ admits a *one-sided-error* heuristic algorithm, then $\mathsf{DistPH}$ admits an *errorless* heuristic algorithm. Here, $\mathsf{co}\mathfrak{C}$ denotes the complement of $\mathfrak{C}$ for a class $\mathfrak{C}$.

7. $\mathsf{coMINKT}^{\mathsf{PH}} \times \mathsf{PSamp} \subseteq \mathsf{Avg}^1_{1-n^{-c}}\mathsf{P}$ *for some constant* $c > 0$.

[9] The standard techniques of error-correcting codes yield such an equivalence for high complexity classes such as $\mathsf{PSPACE}$ and $\mathsf{EXP}$. For example, two-sided-error average-case and worst-case complexity of $\mathsf{PSPACE}$ are equivalent [STV01, TV07], and so is the one-sided-error average-case complexity. However, the proof technique is not applicable to $\mathsf{PH}$ [Vio05]. We also mention that there is a simple argument that works if the failure probability is small (cf. Appendix A).

[10] We mention that it is easy to construct an *artificial* $\mathsf{DistPH}$-complete family of distributional problems [SY96].

Next, we explain how to resolve the issue of monotonicity. As mentioned before, the meta-complexity of $\mathrm{GapMINKT}^A$ is not necessarily monotone increasing with respect to $A$. In our previous work [Hir20a], we introduced the notion of "non-disjoint" promise problems so that the monotonicity can be incorporated into the definition of a problem itself.

Specifically, let $\mathrm{Gap}(\mathrm{K}^{\mathsf{PH}}\text{ vs }\mathrm{K})$ denote (a family of) the promise problems whose YES instances are those of $\mathrm{GapMINKT}^{\mathsf{PH}}$ and NO instances are those of GapMINKT (see Definition 8.3 for a precise definition). This is not a standard promise problem in the sense that, under the plausible assumption that $\mathsf{E}^{\mathsf{NP}} \neq \mathsf{E}$, there exists an instance that is simultaneously a YES and NO instance, and thus $\mathrm{Gap}(\mathrm{K}^{\mathsf{PH}}\text{ vs }\mathrm{K})$ is not a disjoint pair of languages; in this case, any algorithm—not only a polynomial-time algorithm but also literally *any* algorithm—cannot solve $\mathrm{Gap}(\mathrm{K}^{\mathsf{PH}}\text{ vs }\mathrm{K})$. Nevertheless, under the assumption that $\mathsf{DistPH} \subseteq \mathsf{AvgP}$, there exists a polynomial-time algorithm that solves the "non-disjoint" promise problem.

*8.* $\mathrm{Gap}(\mathrm{K}^{\mathsf{PH}}\text{ vs }\mathrm{K}) \in \mathsf{P}$.

The mathematical properties of the meta-complexity of $\mathrm{Gap}(\mathrm{K}^A\text{ vs }\mathrm{K})$ are better and more intuitive than $\mathrm{GapMINKT}^A$. For example, it is not hard to see that there is a many-one reduction from $\mathrm{Gap}(\mathrm{K}^A\text{ vs }\mathrm{K})$ to $\mathrm{Gap}(\mathrm{K}^B\text{ vs }\mathrm{K})$ for any oracles $A \leq_{\mathrm{T}}^p B$ (cf. Lemma 8.14), which serves as a key property for proving the monotonicity of meta-complexity (Theorem 1.13). Moreover, the identity map reduces $\mathrm{GapMINKT}^{\mathsf{PH}}$ to $\mathrm{Gap}(\mathrm{K}^{\mathsf{PH}}\text{ vs }\mathrm{K})$ (cf. Fact 8.4), and thus the latter problem is a harder problem, which explains the implication from Item 8 to 2.

The question is—how can we show that there exists a *polynomial-time algorithm* that can solve the "non-disjoint" promise problem which we believe *no algorithm* can solve? A short answer is that $\mathrm{MINKT}^{\mathsf{PH}}$ is inherently a *meta-computational* problem that encodes a computation as its instance.[11] This enables us to show that, under the assumption that $\mathrm{GapMINKT}^{\mathsf{PH}} \in \mathsf{P}$, for any oracle $A \in \mathsf{PH}$, there exists a polynomial $\tau_A$ such that $\mathrm{K}^{\tau_A(|x|,t)}(x) \leq \mathrm{K}^{t,A}(x) + \log \tau_A(|x|,t)$ for any $x \in \{0,1\}^*$ and any $t \in \mathbb{N}$, in which case $\mathrm{Gap}(\mathrm{K}^{\mathsf{PH}}\text{ vs }\mathrm{K})$ is indeed a *disjoint* promise problem.

One of the key components of the proof is to bridge the gap from the one-sided-error average-case complexity of $\mathsf{coMINKT}^{\mathsf{PH}} \times \mathsf{PSamp}$ to the worst-case meta-complexity of $\mathrm{Gap}(\mathrm{K}^{\mathsf{PH}}\text{ vs }\mathrm{K})$, that is, the implication from Item 7 to Item 8. The gap can be closed by using the proof techniques of the *non-black-box* worst-case-to-average-case reductions of [Hir18, Hir20a]. Unfortunately, the previous worst-case-to-average-case reductions are not error-tolerant, and require the success probability of a one-sided-error heuristic algorithm to be at least $1 - 1/\mathsf{poly}(n)$. We need to reduce the requirement of the success probability to $1/\mathsf{poly}(n)$.

One of the technical contributions of this work is to make the reductions error-tolerant. The main bottleneck of the previous reductions is the existence of the time parameter $t$: In the previous reductions, an instance $(x, 1^t, 1^s)$ was reduced to some instance $(x', 1^{t'}, 1^{s'})$, where $t' = \mathsf{poly}(n,t)$ and $n = |x|$. Because we require solving $\mathrm{GapMINKT}^{\mathsf{PH}}$ for *every* time parameter $t \in \mathbb{N}$, it was also required that a heuristic algorithm solves $(x', 1^{t'}, 1^{s'})$ for *every* time parameter $t'$, which can be ensured if the success probability of the heuristic algorithm is assumed to be at least $1 - 1/\mathsf{poly}(n)$.

A new insight of this work is that the time bound can be fixed to $t := n^\gamma$, where $\gamma > 0$ is an arbitrary constant and $n$ is the length of $x$. For a function $t\colon \mathbb{N} \to \mathbb{N}$, let $\mathrm{GapMINKT}^{\mathsf{PH}}[t = t(n)]$ denote the version of $\mathrm{GapMINKT}^{\mathsf{PH}}$, where the time bound is fixed to $t(|x|)$ on input $(x, 1^s)$. The following is equivalent, for any constant $\gamma > 0$.

*9.* $\mathrm{GapMINKT}^{\mathsf{PH}}[t = n^\gamma] \in \mathsf{P}$.

[11] More specifically, the instances encode some relationships among complexity classes. See Section 2.1.3.

For example, one can regard $\mathrm{GapMINKT}^{\mathsf{PH}}[t = n^{1/10}]$ or $\mathrm{GapMINKT}^{\mathsf{PH}}[t = n^{100}]$ as a problem that characterizes the average-case complexity of $\mathsf{PH}$. We mention in passing that $\mathrm{GapMINKT}^{\mathsf{PH}}[t = n^{\gamma}]$ for $\gamma \in (0,1)$ can be regarded as a *sublinear*-time-bounded Kolmogorov complexity, which is reminiscent of MKTP [ABK+06b, AHK17]. Here, MKTP is the problem of, given an input $x$, computing the trade-off $\mathrm{KT}(x) := \min\{\, \mathrm{K}^t(x) + t \mid t \in \mathbb{N} \,\}$ between a description length and a (sublinear-)time bound.

We have explained the equivalence that connects average-case complexity and meta-complexity of time-bounded Kolmogorov complexity. Our equivalence even extends to the non-existence of a hitting set generator, which is one of the fundamental notions of complexity theory. Recall that a family of functions $H = \{H_n : \{0,1\}^{s(n)} \to \{0,1\}^n\}_{n\in\mathbb{N}}$ is said to be a *hitting set generator* (HSG) secure against a complexity class $\mathfrak{C}$ if, for every $C \in \mathfrak{C}$, for infinitely many $n \in \mathbb{N}$, $\Pr_{w\sim\{0,1\}^n}[C(w) = 1] \geq 1/4$ implies that $C(H_n(z)) = 1$ for some $z \in \{0,1\}^{s(n)}$.

As observed in [HW20], it is not hard to see that the existence of a $\mathsf{PH}$-computable hitting set generator implies $\mathsf{DistPH} \not\subseteq \mathsf{AvgP}$. Surprisingly, we establish the converse direction.

10. *There exists no hitting set generator* $H = \{H_n : \{0,1\}^{n-1} \to \{0,1\}^n\}_{n\in\mathbb{N}}$ *computable in polynomial time with* $\mathsf{PH}$ *oracle that is secure against* $\mathsf{P}$*, and* $\mathsf{P} = \mathsf{ZPP}$*.*[12]

We note that the notion of hitting set generator considered here is *cryptographic* as opposed to *complexity-theoretic*. The latter notion is suitable for derandomizing one-sided-error randomized algorithms, and allows the computational resource for computing a hitting set generator to be larger than its adversary. In contrast, we require that a hitting set generator $H$ is computable in a *fixed polynomial time* (with $\mathsf{PH}$ oracle) and $H$ is secure against an *arbitrary polynomial-time* adversary.

One of the central questions about cryptographic hitting set generators is whether one can extend its seed. It is well known that a cryptographic pseudorandom generator $G = \{G_n : \{0,1\}^{n-1} \to \{0,1\}^n\}_{n\in\mathbb{N}}$ that extends its seed by one bit can be stretched to $\mathsf{poly}(n)$ bits. However, the corresponding question on a hitting set generator is open, as raised by Rudich [Rud97]. We make the first progress towards resolving the question, by showing that the seed of a $\mathsf{PH}$-computable HSG can be extended by 1 bit (¬Item 10) if and only if it can be extended by $O(\log n)$ bits (¬Item 11).

11. *For some constant* $c > 0$*, there exists no* $\mathsf{PH}$*-computable hitting set generator* $H = \{H_n : \{0,1\}^{n-c\log n} \to \{0,1\}^n\}_{n\in\mathbb{N}}$ *that is secure against* $\mathsf{P}$*, and* $\mathsf{P} = \mathsf{ZPP}$*.*

Another natural question regarding a polynomial-time-computable HSG is whether it is equivalent to a sublinear-time-computable HSG. Specifically, for any constant $\gamma > 0$, we say that a HSG $H = \{H_n : \{0,1\}^{s(n)} \to \{0,1\}^n\}_{n\in\mathbb{N}}$ is computable in time $n^{\gamma}$ if, given random access to $z \in \{0,1\}^{s(n)}$ and an index $i \in [n]$, one can compute the $i$th bit of $H_n(z)$ in time $n^{\gamma}$. We show that the following is equivalent for any constant $\gamma > 0$.

12. *For some constant* $c > 0$*, for any constant* $k \in \mathbb{N}$*, there exists no hitting set generator* $H = \{H_n : \{0,1\}^{n-c\log n} \to \{0,1\}^n\}_{n\in\mathbb{N}}$ *computable in time* $n^{\gamma}$ *with* $\Sigma_k\mathsf{SAT}$ *oracle that is secure against* $\mathsf{P}$*, and* $\mathsf{P} = \mathsf{ZPP}$*.*

For example, the non-existence of a $\mathsf{PH}$-oracle $n^{1/10}$-computable HSG (for $\gamma := 1/10$) and that of a $\mathsf{PH}$-oracle $n^{100}$-computable HSG (for $\gamma := 100$) are equivalent. This is a rather counterintuitive

[12] For some technical reason, we include in Item 10 the mild derandomization assumption that $\mathsf{P} = \mathsf{ZPP}$. In particular, Item 10 is equivalent to Items 1 to 9 under the assumption that $\mathsf{P} = \mathsf{ZPP}$.

result: Naively, one can imagine that, if the time bound $n^{1/10}$ is increased to $n^{100}$, more strings can be computed, and thus a hitting set generator should become more secure. As a consequence, one might guess that Item 12 should not be equivalent to the non-existence of $\mathsf{PH}$-computable hitting set generators. This intuition turns out to be not correct.

Instead, it is instructive to consider Item 12 as a HSG analogue of the pseudorandom function generator construction of Goldreich, Goldwasser, and Micali [GGM86], from which it follows that any $\mathsf{poly}(n)$-time computable PRG can be converted to an $n^{\gamma}$-time computable PRG, where $n$ denotes the output length of PRGs. ◊

Fig. 2 summarizes some important statements and our proof strategies. On the top half of the figure are the statements on average-case complexity. On the bottom half of the figure are the statements on worst-case meta-complexity. The essential steps in our proof are to connect these fundamentally different statements.

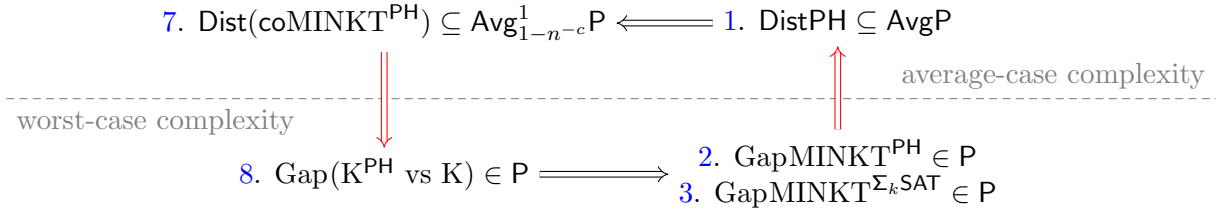


Figure 2: Some statements in the equivalence of the main theorem. The main technical implications are highlighted in red.

One crucial step that brings us from the average-case-complexity world to the worst-case meta-complexity world is the following, which provides the non-black-box error-tolerant worst-case-to-average-case reduction and improves [Hir18, Hir20a].

**Theorem 1.15** (Item 7 ⇒ 8)**.** *Let $c > 0$ be any constant and $A$ be any* NP*-hard oracle.*[13] *If* $\{\mathsf{coMINKT}^A\} \times \mathsf{PSamp} \subseteq \mathsf{Avg}^1_{1-n^{-c}}\mathsf{P}$*, then* $\mathrm{Gap}(\mathrm{K}^A \text{ vs } \mathrm{K}) \in \mathsf{P}$.

Due to the barrier of Bogdanov and Trevisan [BT06b], this step cannot be regarded as a (black-box) worst-case-to-average-case reduction (see [Hir18, HW20] for detailed discussion); thereby it crosses the boundary from the average-case world to the worst-case meta-complexity world.

Another crucial step that brings us from the worst-case meta-complexity world to the average-complexity world is the following, which establishes "DistPH-hardness" of GapMINKT$^{\mathsf{PH}}$ by building on the ideas developed in [Hir20c, Hir20b].

**Theorem 1.16** (Item 3 ⇒ 1)**.** *Let $A$ be any $\Sigma^{\mathrm{p}}_k$-hard problem for some $k \in \mathbb{N}$. If* $\mathrm{GapMINKT}^A \in \mathsf{P}$*, then* $\mathsf{Dist}\Sigma^{\mathrm{p}}_k \subseteq \mathsf{AvgP}$.

Since our hardness amplification theorem for $\mathsf{PH}$ (Theorem 1.6) is a statement purely on average-case complexity, it is natural to ask whether we can simplify our proof to provide a purely average-case complexity-theoretic argument. Note that Theorem 1.16 provides an "average-case-to-worst-case" reduction. If we could interpret this reduction as an average-case-to-average-case reduction,

[13] The NP-hardness of $A$ is used to construct an explicit pseudorandom generator of logarithmic seed length secure against linear-sized circuits. In particular, under the plausible assumption that $\mathsf{E} \not\subseteq \bigcap_{\epsilon>0} \mathsf{i.o.SIZE}(2^{\epsilon n})$ [IW97], Theorem 1.15 holds for any oracle $A$.

then we would have obtained an average-case complexity-theoretic proof easily. Surprisingly, the reduction of Theorem 1.16 *cannot* be regarded as an average-case-to-average-case reduction for any $k \geq 2$, and it is essential to cross the boundary from the worst-case meta-complexity world to the average-case world. It is this interplay between average-case complexity and worst-case meta-complexity that enables resolving the fundamental open questions. The details will be explained in Section 2.2 under the name of the "$\mathsf{S}_2^{\mathsf{p}}$-barrier".

Both of Theorems 1.15 and 1.16 make use of the existence of a (complexity-theoretic) pseudo-random generator. In fact, one of our main technical contributions is to prove $\mathsf{P} = \mathsf{BPP}$ under the assumptions that any one of Items 1 to 12 holds. For example:

**Theorem 1.17** ($\mathsf{BPP}$-hardness)**.** *Let $A$ be any $\mathsf{NP}$-hard oracle. If $\mathrm{GapMINKT}^A \in \mathsf{P}$, then $\mathsf{P} = \mathsf{BPP}$.*

Fig. 3 summarizes the relationship among statements on different levels of $\mathsf{PH}$.

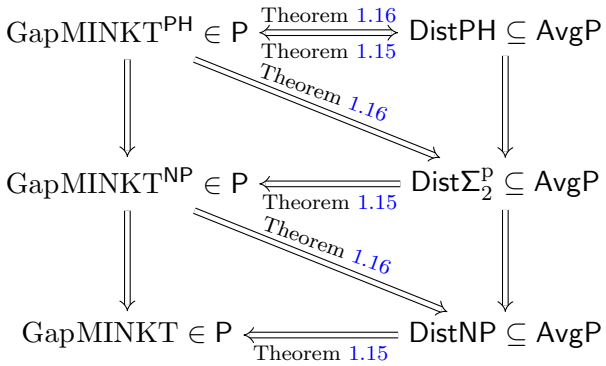


Figure 3: The relationships between $\mathrm{GapMINKT}^{\mathfrak{C}}$ for $\mathfrak{C} \in \{\mathsf{P}, \mathsf{NP}, \mathsf{PH}\}$ and average-case complexity.

### 1.3.1 Meta-Complexity is Indispensable for Average-Case Complexity

Our results have important consequences to the open questions mentioned before.

As mentioned in Open Question 1.10, the fundamental open question of GapMINKT is to prove its "$\mathsf{NP}$-completeness". In our previous work [Hir18, Hir20a], we showed that Open Question 1.10 is sufficient for equating the worst-case and average-case complexity of $\mathsf{NP}$ (i.e., Open Question 1.10 implies Open Question 1.1). The results of [Hir18] overcame the significant barrier of Bogdanov and Trevisan [BT06b]: No non-adaptive black-box reduction can reduce $\mathsf{NP}$-complete problems to $\mathsf{DistNP}$ (unless $\mathsf{PH}$ collapses) nor reduce GapMINKT to $\mathsf{DistNP}$ (unless $\mathrm{GapMINKT} \in \mathsf{coNP/poly}$); in contrast, the proof techniques of [Hir18] are non-black-box. However, it is possible that *adaptive* reductions can also bypass the barrier of [BT06b], and it was not clear at all whether proving $\mathsf{NP}$-hardness of meta-computational problems is really necessarily for resolving Open Question 1.1.

The results of this work indicate that we cannot resolve Open Question 1.1 without resolving "$\mathsf{NP}$-hardness" of $\mathrm{GapMINKT}^{\mathsf{PH}}$ (Open Question 1.11); therefore, *meta-complexity is indispensable for studying average-case complexity*. To summarize, using informal notations such as $\mathsf{NP} \leq \mathsf{DistPH}$ (meaning that $\mathsf{DistPH} \subseteq \mathsf{AvgP} \implies \mathsf{NP} = \mathsf{P}$), we have the following relationships among open questions.

**Corollary 1.18.** *Open Question 1.10* ($\mathsf{NP} \leq \mathrm{GapMINKT}$) $\implies$ *Open Question 1.1* ($\mathsf{NP} \leq \mathsf{DistNP}$) $\implies$ *Open Question 1.5* ($\mathsf{NP} \leq \mathsf{DistPH}$) $\iff$ *Open Question 1.11* ($\mathsf{NP} \leq \mathrm{GapMINKT}^{\mathsf{PH}}$).

### 1.3.2 A New Approach Towards Hardness Amplification for NP

There are oracles relative to which Open Questions 1.1 and 1.10 do not hold, as constructed by Ko [Ko91] and Impagliazzo [Imp11], respectively. Therefore, we need to develop a non-relativizing proof technique in order to resolve these open questions. In light of this, we propose an open question that is less challenging but still has an important consequence:

**Open Question 1.19** ("DistNP-hardness" of GapMINKT)**.** Does $\mathrm{GapMINKT} \in \mathsf{P}$ implies $\mathsf{DistNP} \subseteq \mathsf{AvgP}$?

Open Question 1.19 provides a completely new approach towards improving the hardness amplification result of Bogdanov and Safra [BS07] from $1/(\log n)^{1/10}$ to $1/\mathsf{poly}(n)$. Specifically:

**Corollary 1.20** (of Theorem 1.15)**.** *Open Question 1.19 implies Open Question 1.2 (i.e., the hardness amplification theorem for* NP*).*[14]

Note that Theorem 1.16 shows "DistNP-hardness" of $\mathrm{GapMINKT}^{\mathsf{NP}}$; Open Question 1.19 asks whether the NP-oracle can be eliminated. It should be also noted that Open Question 1.19 would classify the complexity of GapMINKT as a "DistNP-complete" problem in light of Theorem 1.15.

## 1.4 Related Work

We present several previous works that are closely related to this work as well as impacts of our results to theirs.

### 1.4.1 From Average-Case Complexity to Meta-Complexity

In the area of meta-complexity, the proof techniques of average-case complexity have been often exploited. Using random self-reducibility and downward self-reducibility of some PSPACE-complete problem [TV07], Allender, Buhrman, Koucký, van Melkebeek, Ronneburger [ABK+06b] showed that $\mathsf{PSPACE} \subseteq \mathsf{ZPP}^{\mathrm{MCSP}^{\mathsf{PSPACE}}}$, and the same proof technique shows that $\mathsf{PSPACE} \subseteq \mathsf{ZPP}^{\mathrm{GapMINKT}^{\mathsf{PSPACE}}}$. Combining their results with the BPP-hardness (Theorem 1.17), we immediately obtain "PSPACE-completeness" of $\mathrm{GapMINKT}^{\mathsf{PSPACE}}$ under deterministic reductions.

**Corollary 1.21.** $\mathrm{GapMINKT}^{\mathsf{PSPACE}} \in \mathsf{P}$ *if and only if* $\mathsf{PSPACE} = \mathsf{P}$.

*Proof Sketch.* If $\mathrm{GapMINKT}^{\mathsf{PSPACE}} \in \mathsf{P}$, [ABK+06b] implies $\mathsf{PSPACE} = \mathsf{ZPP}$. By Theorem 1.17, we also have $\mathsf{ZPP} \subseteq \mathsf{BPP} = \mathsf{P}$. □

Impagliazzo, Kabanets, and Volkovich [IKV18] generalized the result of [ABK+06b] to $\mathfrak{C} \subseteq \mathsf{BPP}^{\mathrm{GapMINKT}^{\mathfrak{C}}}$ for any $\mathfrak{C} \in \{\oplus\mathsf{P}, \mathsf{P}^{\#\mathsf{P}}, \mathsf{PP}\}$. As in Corollary 1.21, Theorem 1.17 enables improving their hardness results under randomized reductions to "deterministic reductions."

**Corollary 1.22.** *Let* $\mathfrak{C} \in \{\mathsf{BPP}^{\oplus\mathsf{P}}, \mathsf{P}^{\#\mathsf{P}}, \mathsf{PP}\}$. *If* $\mathrm{GapMINKT}^{\mathfrak{C}} \in \mathsf{P}$, *then* $\mathfrak{C} = \mathsf{P}$.

[14] To be more precise, it is a corollary of Theorem 6.5 and Lemma 8.9.

*Proof Sketch.* Since $\mathfrak{C}$ includes $\mathsf{NP}$ (where $\mathsf{NP} \subseteq \mathsf{BPP}^{\oplus\mathsf{P}}$ is due to [VV86]),[15] we can apply Theorem 1.17 and obtain $\mathsf{P} = \mathsf{BPP}$; thus, [IKV18] implies $\mathfrak{C} \subseteq \mathsf{BPP} = \mathsf{P}$. □

### 1.4.2 From Kolmogorov Complexity to Average-Case Complexity

Previously, Kolmogorov complexity *(instead of its meta-complexity)* was considered as a fundamental tool for analyzing average-case complexity. For example, Li and Vitányi [LV92] used Kolmogorov complexity to define a (not computable) distribution under which the average-case and worst-case complexity are equivalent.

Antunes and Fortnow [AF09] characterized the running time of average-case algorithms by using the notion of *computational depth*. The computational depth (with time bound $t$) of a string $x$ is defined as $\mathrm{cd}^t(x) := \mathrm{K}^t(x) - \mathrm{K}(x)$, whose notion was introduced by Antunes, Fortnow, van Melkebeek, and Vinodchandran [AFvMV06]. Under the assumption that exponential time is not infinitely often in sub-exponential space, it was shown in [AF09] that, for all polynomial $p$, the running time of $A$ is bounded by $2^{\mathrm{cd}^{p(|x|)}(x)+O(\log|x|)}$ for any input $x$ if and only if $A$ runs in average-case polynomial time with respect to any distribution $\mu \in \mathsf{PSamp}$.

We emphasize the fundamental differences between these previous results and this work. Our work characterizes average-case complexity via the *meta-complexity* of time-bounded Kolmogorov complexity, whereas the previous results characterize average-case complexity via Kolmogorov complexity itself. It should be also noted that the result of [AF09] is a conditional result, whereas our equivalence is unconditional, for which we make significant technical contributions.

### 1.4.3 Cryptographic Hitting Set Generator

Santhanam [San20] posed the Universality Conjecture, under which a "succinct" hitting set generator can be extended arbitrary. The conjecture cannot be refuted unless one-way functions fail to exist, and, at the same time, its solvability remains unclear. It is left as an interesting research direction to make progress towards the Universality Conjecture using the proof techniques behind Theorem 1.14, which shows that a $\mathsf{PH}$-computable hitting set generator can be slightly extended.

### 1.4.4 Hardness Amplification

We mention some previous works on hardness amplification. Impagliazzo, Jaiswal, Kabanets, and Wigderson [IJK09, IJKW10] showed a uniform version of Yao's XOR lemma; in particular, it was shown that, if $\mathsf{P}_{\|}^{\mathsf{NP}} \times \{\mathcal{U}\} \subseteq \mathsf{Heur}_{1/2+1/\mathsf{poly}(n)}\mathsf{BPP}$, then $\mathsf{P}_{\|}^{\mathsf{NP}} \times \{\mathcal{U}\} \subseteq \mathsf{Heur}_{1/\mathsf{poly}(n)}\mathsf{BPP}$. Bogdanov and Safra [BS07] proved a non-uniform and errorless version of Yao's XOR lemma, and showed that if $\mathfrak{C} \times \{\mathcal{U}\} \subseteq \mathsf{Avg}_{1-1/\mathsf{poly}(n)}\mathsf{P}/\mathsf{poly}$ then $\mathfrak{C} \times \{\mathcal{U}\} \subseteq \mathsf{AvgP}/\mathsf{poly}$ for any class $\mathfrak{C}$ closed under taking XORs.

Note that, if there were a *uniform and errorless* version of Yao's XOR lemma, it would have provided an alternative proof of the $\mathsf{PH}$ analogue of Open Question 1.2 (i.e., the hardness amplification theorem for $\mathsf{PH}$ against uniform algorithms). However, it was noted in [Wat15] that the proof techniques of [IJK09, IJKW10] "do not seem to apply to the errorless setting."

[15] Since it is not known whether $\mathsf{NP} \subseteq \oplus\mathsf{P}$, we do not know whether Corollary 1.22 holds for $\mathfrak{C} = \oplus\mathsf{P}$.

## 2 Proof Techniques

We present more details of our proof techniques. The following are highlights of our new technical contributions.

**Derandomization (BPP-hardness)**
If $\mathrm{GapMINKT}^{\mathsf{NP}} \in \mathsf{P}$, then $\mathsf{P} = \mathsf{BPP}$ (Theorem 1.17).

**DistPH-hardness**
If $\mathrm{GapMINKT}^{\mathsf{PH}} \in \mathsf{P}$, then $\mathsf{DistPH} \subseteq \mathsf{AvgBPP} = \mathsf{AvgP}$ (Theorem 1.16).

**Error-tolerant worst-case-to-average-case reduction**
If $\mathrm{coMINKT}^{\mathsf{PH}} \times \mathsf{PSamp} \subseteq \mathsf{Avg}^{1}_{1-1/\mathsf{poly}(n)}\mathsf{P}$, then $\mathrm{Gap}(\mathrm{K}^{\mathsf{PH}} \text{ vs } \mathrm{K}) \in \mathsf{P}$ (Theorem 1.15).

These results are explained in the following subsections.

### 2.1 Derandomization

A large disparity between the statements that $\mathrm{GapMINKT}^{\mathsf{PH}} \in \mathsf{P}$ and that $\mathsf{DistPH} \subseteq \mathsf{AvgP}$ is that the latter statement implies $\mathsf{P} = \mathsf{BPP}$ [BFP05] whereas it was not known whether the former implies $\mathsf{P} = \mathsf{BPP}$. In order to establish the equivalence, we need to develop a proof technique that yields the same consequence.

**Theorem 1.17.** *If* $\mathrm{GapMINKT}^{\mathsf{NP}} \in \mathsf{P}$, *then* $\mathsf{P} = \mathsf{BPP}$.

#### 2.1.1 Background

Before explaining its proof, we emphasize again that Theorem 1.17 is a highly non-trivial result. If $\mathrm{GapMINKT}^{\mathsf{NP}}$ is $\mathsf{NP}$-hard, then we will obtain $\mathsf{P} = \mathsf{NP}$ (under the assumption that $\mathrm{GapMINKT}^{\mathsf{NP}} \in \mathsf{P}$), and consequently $\mathsf{BPP} \subseteq \mathsf{NP}^{\mathsf{NP}} \subseteq \mathsf{P}$. However, $\mathrm{GapMINKT}^{\mathsf{NP}}$ asks for the meta-complexity of minimizing an $\mathsf{NP}$-oracle program, and the relationship between the "plain complexity" of $\mathsf{NP}$ and the meta-complexity of $\mathrm{GapMINKT}^{\mathsf{NP}}$ is not obvious at all. Nevertheless, we were able to "extract" the plain complexity of $\mathsf{BPP}$ from the meta-complexity of $\mathrm{GapMINKT}^{\mathsf{NP}}$ via a *deterministic reduction*. In fact, Theorem 1.17 is one of the most technical components in this paper.

There is a significant technical barrier that we need to overcome. It has been deemed that deterministic reductions are too restricted notions to extract plain complexity from meta-complexity. For example, Buhrman and Mayordomo [BM97] showed that $\mathsf{EXP} \neq \mathsf{P}^{R_{\mathrm{K}^{t(n)}}}$, where $t(n) := 2^{n^2}$ and $R_{\mathrm{K}^{t(n)}}$ denotes the set $\{\, x \in \{0,1\}^* \mid \mathrm{K}^{t(|x|)}(x) \geq |x| \,\}$ of time-bounded-Kolmogorov-random strings. As a consequence, we cannot prove $\mathsf{ZPP} \leq^{p}_{\mathrm{T}} R_{\mathrm{K}^{t(n)}}$ without resolving the notorious open question $\mathsf{EXP} \neq \mathsf{ZPP}$. Similarly, Murray and Williams [MW17] showed that $\mathsf{ZPP} \leq^{p}_{\mathrm{m}} \mathrm{MCSP}$ implies that $\mathsf{EXP} \neq \mathsf{ZPP}$.[16]

It is instructive to recall the ideas behind [BM97, MW17] in order to explain why deterministic reductions are significantly restricted. The main reason is that a deterministic reduction cannot produce any string with high Kolmogorov complexity. To be more specific, let $1^n$ be an input to a (nonadaptive) deterministic polynomial-time reduction $M$, and consider an arbitrary query

[16] The relationship between [BM97] and [MW17] can be found in [HW16]. See also [SS20] for recent results.

$q$ that $M$ can yield on input $1^n$. The query $q$ can be described by using the program $M$, the input length $n$, and the index of the queries, whose description length is $O(\log n)$; thus, we always have $\mathrm{K}^t(q) = O(\log n)$ for a sufficiently large $t$, and thus $q \notin R_{\mathrm{K}^{t(n)}}$ for any query $q$ such that $|q| > O(\log n)$. This means that any nonadaptive deterministic polynomial-time reduction cannot extract any useful information on whether $q \in R_{\mathrm{K}^{t(n)}}$ for any long query $q$.

Based on this intuition, Allender, Buhrman, and Koucký [ABK06a] and Allender [All12] proposed several conjectures on the limits of hardness results of (resource-unbounded) Kolmogorov complexity, which, if true, would establish severe limits that plain complexity cannot be extracted from meta-complexity.[17]

In our previous works [Hir20c, Hir20b], we presented unexpected hardness results that refute these conjectures under plausible complexity-theoretic assumptions. We prove Theorem 1.17 based on the proof ideas developed in [Hir18, Hir20a, Hir20c, Hir20b]. In order to improve the efficiency of reductions, we will require a new black-box hitting set generator construction whose reconstruction algorithm uses few random bits.

### 2.1.2 Proof Outline

We outline the proof of Theorem 1.17. The starting point is the result of Buhrman, Fortnow, and Pavan [BFP05] that shows that, if $\mathsf{DistNP} \subseteq \mathsf{AvgP}$, then $\mathsf{P} = \mathsf{BPP}$. In fact, their result is stronger: There exists an explicit (complexity-theoretic) pseudorandom generator $G = \{G_n : \{0,1\}^{O(\log n)} \to \{0,1\}^n\}_{n\in\mathbb{N}}$ secure against linear-sized circuits. Note that, once a pseudorandom generator with logarithmic seed length is constructed, we can derandomize any $\mathsf{BPP}$-computation by trying all the possible seeds $z \in \{0,1\}^{O(\log n)}$ and using $G_n(z)$ as the source of randomness. We will also show (and need) a stronger result that $\mathrm{GapMINKT}^{\mathsf{NP}} \in \mathsf{P}$ implies the existence of a nearly optimal pseudorandom generator.

We review the proof of [BFP05] below. Their proof relies on the following two results:

1. If $\mathsf{DistNP} \subseteq \mathsf{AvgP}$ then $\mathsf{E} = \mathsf{NE}$ [BCGL92].
2. If $\mathsf{DistNP} \subseteq \mathsf{AvgP}$ then $\mathsf{pr\text{-}MA} = \mathsf{NP}$ [KS04].

It was shown in [BFP05] that these results imply the existence of a nearly optimal pseudorandom generator. (Proof Sketch: $\mathsf{E} \subseteq \mathsf{i.o.SIZE}(2^{o(n)})$ implies $\mathsf{E} \subseteq \mathsf{i.o.MATIME}(2^{o(n)}) = \mathsf{i.o.NTIME}(2^{o(n)})$, which contradicts $\mathsf{E} = \mathsf{NE}$ and the deterministic time hierarchy theorem.)

Our plan is to replace the assumption that $\mathsf{DistNP} \subseteq \mathsf{AvgP}$ of Items 1 and 2 with one that $\mathrm{GapMINKT}^{\mathsf{NP}} \in \mathsf{P}$. It is not hard to see that $\mathrm{GapMINKT}^{\mathsf{NP}} \in \mathsf{P}$ implies $\mathsf{pr\text{-}MA} = \mathsf{NP}$, which establishes the $\mathrm{GapMINKT}^{\mathsf{NP}}$ analogue of Item 2. Indeed, as shown in [ABK+06b], in order to nondeterministically derandomize $\mathsf{pr\text{-}MA}$, one can guess a hard function $f$, verify the hardness of $f$ using GapMINKT, and derandomize $\mathsf{pr\text{-}MA}$ by using standard pseudorandom generator constructions $G^f$ (such as [NW94, IW97]) based on a hard function $f$.

In contrast, the $\mathrm{GapMINKT}^{\mathsf{NP}}$ analogue of Item 1 is much more technically difficult. At the core of Theorem 1.17 is the following new hardness result under deterministic reductions.

**Theorem 5.1.** *$\mathsf{E}^{\mathsf{NP}}$ is reducible to $\mathrm{GapMINKT}^{\mathsf{NP}}$ via a deterministic nonadaptive $\mathsf{E}$-reduction. In particular, $\mathsf{E}^{\mathsf{NP}} = \mathsf{E}$ holds if $\mathrm{GapMINKT}^{\mathsf{NP}} \in \mathsf{P}$.*

[17] Allender, Friedman, and Gasarch [AFG13] made remarkable progress by showing that any computable language that is reducible to the set of prefix-free Kolmogorov-random strings under deterministic polynomial-time nonadaptive reductions (irrespective of the choice of prefix-free universal Turing machines) is in $\mathsf{PSPACE}$.

Interestingly, this result bypasses the technical barrier mentioned before: The reduction of Theorem 5.1 extracts useful information from short queries to GapMINKT$^{\mathsf{NP}}$ oracle. On the other hand, it is difficult to prove a similar hardness result for problems on "subpolynomial"-time-bounded Kolmogorov complexity, such as MKTP$^{\mathsf{NP}}$ or MCSP$^{\mathsf{NP}}$.

**Proposition 2.1** (see [Hir20b])**.** *If* $\mathsf{E}^{\mathsf{NP}}$ *is reducible to either* MCSP$^{\mathsf{NP}}$ *or* MKTP$^{\mathsf{NP}}$ *via a deterministic nonadaptive* $\mathsf{E}$*-reduction, then* $\mathsf{EXP} \neq \mathsf{ZPP}$*.*

### 2.1.3 Meta-Computational View of $\mathsf{E}^{\mathsf{NP}}$ versus $\mathsf{E}$

Below, we explain the idea of the new $\mathsf{E}^{\mathsf{NP}}$-hardness of GapMINKT$^{\mathsf{NP}}$ under $\mathsf{E}$-reductions (Theorem 5.1). For simplicity, we explain the idea of the proof that shows $\mathsf{E}^{\mathsf{NP}} = \mathsf{E}$ under the assumption that GapMINKT$^{\mathsf{NP}} \in \mathsf{P}$.

What we aim to understand is the relationship between the two complexity classes: One is $\mathsf{E}^{\mathsf{NP}}$ and the other is $\mathsf{E}$. It is useful to view the relationship from the *meta-computational perspective*. Specifically, stepping back, let us rephrase the question of $\mathsf{E}^{\mathsf{NP}}$ versus $\mathsf{E}$ as the question of time-bounded Kolmogorov complexity. Using the fact that $\mathsf{E}^{\mathsf{NP}} = \mathsf{E}$ if and only if $\mathsf{E}^{\mathsf{NP}} \subseteq \mathsf{E}/O(n)$ [BH92, Hir15], it is not hard to observe the following.

**Fact 2.2.** *Assume that, for all* $A \in \mathsf{NP}$*, there exists some polynomial* $\tau$ *such that, for any* $t \in \mathbb{N}$ *and any family of strings* $\{x_N \in \{0,1\}^N\}_{N\in\mathbb{N}}$ *with* $\mathrm{K}^{t,A}(x_N) = O(\log N)$*, it holds that* $\mathrm{K}^{\tau(N,t)}(x) = O(\log N)$*. Then,* $\mathsf{E}^{\mathsf{NP}} = \mathsf{E}$*.*

*Proof.* Consider any function $f = \{f_n\}_{n\in\mathbb{N}} \in \mathsf{E}^{\mathsf{NP}}$. Let $x_N \in \{0,1\}^N$ be the truth table of $f_n\colon \{0,1\}^n \to \{0,1\}$, where $n := \log N$. Since $f \in \mathsf{E}^{\mathsf{NP}}$, the truth table $x_N$ can be described by using the description of $N$ in polynomial time with $\mathsf{NP}$ oracle; that is, $\mathrm{K}^{t,A}(x_N) = O(\log N)$ for some $A \in \mathsf{NP}$ and some $t = \mathsf{poly}(N)$. By the assumption, it follows that $\mathrm{K}^{\mathsf{poly}(N)}(x_N) = O(\log N)$; therefore, there exists some description $d_N$ of length $O(\log N)$ such that the universal Turing machine $U^{d_N}$ computes each bit of $x_N$ in time $\mathsf{poly}(N)$. This means that there exists a machine that takes an advice string $d_N$ of length $O(\log N) = O(n)$, runs in time $N^{O(1)} = 2^{O(n)}$, and computes $f_n$, which means that $\mathsf{E}^{\mathsf{NP}} \subseteq \mathsf{E}/O(n)$. □

Fact 2.2 reveals the relationship between the assumption that GapMINKT$^{\mathsf{NP}} \in \mathsf{P}$ and the conclusion that $\mathsf{E}^{\mathsf{NP}} = \mathsf{E}$. Assuming that the *meta-complexity* of GapMINKT$^{\mathsf{NP}}$ is easy, we would like to extract a "plain complexity" statement that $\mathrm{K}^{\mathsf{poly}(|x|,t)}(x) \leq O(\mathrm{K}^{t,\mathsf{NP}}(x))$ for any $x$ such that $\mathrm{K}^{t,\mathsf{NP}}(x) = O(\log |x|)$.

With this view in mind, it turns out that the non-black-box worst-case-to-average-case reduction of [Hir18, Hir20a] is useful.

**Theorem 2.3** ([Hir20a])**.** *Let* $A$ *be any oracle. Assume that there exists an explicit pseudorandom generator* $G = \{G_n : \{0,1\}^{O(\log n)} \to \{0,1\}^n\}_{n\in\mathbb{N}}$ *secure against linear-sized circuits, and that there exists an errorless heuristic polynomial-time algorithm that solves* MINKT$^A$*. Then,* Gap($\mathrm{K}^A$ vs K) $\in \mathsf{P}$*.*

The statement of Theorem 2.3 is purely meta-complexity-theoretic. It establishes the connection between two meta-computational problems MINKT$^A$ and Gap($\mathrm{K}^A$ vs K). Nevertheless, from the "non-disjoint" property of Gap($\mathrm{K}^A$ vs K), we can immediately extract the information on plain complexity: Specifically, the statement that Gap($\mathrm{K}^A$ vs K) $\in \mathsf{P}$ implies that $\mathrm{K}^{\mathsf{poly}(|x|,t)}(x) \leq \mathrm{K}^{t,A}(x) + O(\log(|x| + t))$ for any $x \in \{0,1\}^*$ and $t \in \mathbb{N}$.

A demerit of Theorem 2.3 is that it is a conditional result that relies on the existence of a nearly optimal pseudorandom generator. Recall that we aim to construct a pseudorandom generator in the end; therefore, Theorem 2.3 is clearly not useful for our purpose. Without using a pseudorandom generator, the approximation quality of the non-black-box worst-case-to-average-case reduction of [Hir18] is significantly worse.

**Theorem 2.4** (Implicit in [Hir18, Hir20a])**.** *Let $A$ be any oracle. If* $\mathrm{MINKT}^A \in \mathsf{P}$*, then* $\mathrm{K}^{\mathsf{poly}(|x|,t)}(x) \leq \sigma(|x|, \mathrm{K}^{t,A}(x))$*, where* $\sigma(n,s) := s + O(\sqrt{s}\log n + \log^2 n)$*.*

In particular, we obtain $\mathrm{K}^{\mathsf{poly}(|x|,t)}(x) = O(\log^2 |x|)$ for any $x$ such that $\mathrm{K}^{t,A}(x) = O(\log |x|)$. Using the meta-complexity view of Fact 2.2, it follows from Theorem 2.4 that $\mathsf{E}^{\mathsf{NP}} \subseteq \mathsf{E}/O(n^2)$ if $\mathrm{GapMINKT}^{\mathsf{NP}} \in \mathsf{P}$. Unfortunately, this is not efficient enough for constructing a pseudorandom generator with logarithmic seed length. At the core of Theorem 5.1 is the following improvement.

**Theorem 8.1.** *Let $A$ be an oracle. Assume either* $\mathrm{GapMINKT}^A \in \mathsf{P}$ *or* $(\mathsf{coMINKT}^A, \mathcal{D}) \in \mathsf{Avg}^1_{1/4m}\mathsf{P}$ *for any efficiently samplable distribution $\mathcal{D}$.*[18] *Then, there exists a polynomial-time algorithm that solves the following promise problem* $\Pi = (\Pi_{\mathrm{YES}}, \Pi_{\mathrm{NO}})$*.*

$$\Pi_{\mathrm{YES}} := \{\, (x, 1^t, 1^s) \mid \mathrm{K}^{t,A}(x) \leq s \text{ and } s \leq 2^{\log^{1/3}|x|} \,\},$$
$$\Pi_{\mathrm{NO}} := \{\, (x, 1^t, 1^s) \mid \mathrm{K}^{\tau(|x|,t)}(x) > \sigma(|x|, t, s) \text{ and } s \leq 2^{\log^{1/3}|x|} \,\},$$

*where* $\sigma(n,t,s) := 2s + O(\log nt)$ *and $\tau$ is some polynomial.*

To compare Theorem 8.1 with [Hir18], the worst-case to average-case reduction of [Hir18] is a *randomized* reduction that achieves $\sigma(n,s) = s + O(\sqrt{s}\log n + \log^2 n)$. Theorem 8.1 is a *deterministic* reduction, and moreover it achieves a smaller $\sigma$ if $s \leq \log^2 n$.

**Corollary 2.5.** *Under the same assumption of Theorem 8.1,* $\mathrm{K}^{\tau(|x|,t)}(x) \leq 2\mathrm{K}^{t,A}(x) + O(\log |x|)$ *holds for any $t \in \mathbb{N}$ and any $x \in \{0,1\}^*$ such that* $\mathrm{K}^{t,A}(x) \leq 2^{\log^{1/3}|x|}$*.*

*Proof.* Define $s := \mathrm{K}^{t,A}(x)$. Since there exists some algorithm that solves $\Pi = (\Pi_{\mathrm{YES}}, \Pi_{\mathrm{NO}})$, the promise problem must be disjoint; that is, $\Pi_{\mathrm{YES}} \cap \Pi_{\mathrm{NO}} = \varnothing$. Since $(x, 1^t, 1^s) \in \Pi_{\mathrm{YES}}$, we obtain $(x, 1^t, 1^s) \notin \Pi_{\mathrm{NO}}$, from which the result follows. □

This enables us to show that $\mathrm{GapMINKT}^{\mathsf{NP}} \in \mathsf{P}$ implies $\mathrm{K}^{\mathsf{poly}(|x|,t)}(x) = O(\log |x|)$ for any $x$ such that $\mathrm{K}^{t,\mathsf{NP}}(x) = O(\log |x|)$, and thus $\mathsf{E}^{\mathsf{NP}} \subseteq \mathsf{E}$ follows from Fact 2.2.

### 2.1.4 Improved Black-Box Hitting Set Generator Construction

It remains to explain how to prove Theorem 8.1. A tool to connect plain complexity and meta-complexity is a *black-box hitting set generator construction*.

The standard construction of a complexity-theoretic hitting set generator (or a pseudorandom generator of, e.g., [NW94, IW97]) can be regarded as a "black-box" procedure $H$ that takes an arbitrary candidate hard function $f$ and a seed $z$, and outputs a pseudorandom sequence $H(f,z)$. Let us identify a function $f\colon \{0,1\}^{\log n} \to \{0,1\}$ with its truth table $f \in \{0,1\}^n$, and let $H\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ be a black-box hitting set generator construction. Then, the security of a candidate hitting set generator $H(f,\text{-})\colon \{0,1\}^d \to \{0,1\}^m$ is established as follows. There exists

[18] In fact, the former assumption is stronger than the latter in some sense; see Lemma 8.2

some efficient "reconstruction procedure" $R^{(\cdot)}$ associated with $H$ such that, given any function $D\colon \{0,1\}^m \to \{0,1\}$ that *avoids* $H(f,\cdot)$ (which means that $D$ violates the condition of the security of $H(f,\cdot)$), the reconstruction procedure $R^D(\alpha)$ computes $f$ for some advice string $\alpha \in \{0,1\}^a$. In particular, if $f$ is a hard function that cannot be computed by a small circuit, then $D$ cannot be a small circuit. This shows the security of the hitting set generator $H(f,\cdot)$ based on a hard function $f$. It was shown in [Hir20a] that any black-box hitting set generator construction yields a non-black-box worst-case to average-case reduction analogous to Theorem 8.1.

However, the parameters that are required to achieve the approximation quality of Theorem 8.1 are considerably stringent. We need to construct a black-box hitting set generator with advice complexity $a = O(m)$ when $m = O(\log n)$. The main technical component of Theorem 5.1 is to construct such a hitting set generator:

**Theorem 4.3.** *For any sufficiently large $n, m \in \mathbb{N}$ such that $m \leq 2n$, there exists a function $H\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ and a* deterministic *reconstruction procedure $R^{(\cdot)}\colon \{0,1\}^a \to \{0,1\}^n$, where $d = O(\log n + \log^3 m)$ and $a = 2m + O(\log n + \log^3 m)$, such that, for any $x \in \{0,1\}^n$ and any function $D\colon \{0,1\}^m \to \{0,1\}$ that avoids $H(x,\cdot)$, there exists an advice string $\alpha$ such that $R^D(\alpha) = x$. Moreover, $H$ can be computed in time $\mathsf{poly}(n)$ and $R^D$ can be computed in time $\mathsf{poly}(n)$ with oracle access to $D$.*

Using Theorem 4.3, we can prove Theorem 5.1 as follows. Using GapMINKT$^{\mathsf{NP}}$ oracle, we define an efficient algorithm $D$ that avoids $H(f,\cdot)$ for any function $f \in \mathsf{E}^{\mathsf{NP}}$.[19] Theorem 4.3 implies that $f \in \mathsf{E}^D/O(n) = \mathsf{E}/O(n)$. We conclude that $\mathsf{E}^{\mathsf{NP}} \subseteq \mathsf{E}/O(n)$, which is equivalent to $\mathsf{E}^{\mathsf{NP}} = \mathsf{E}$. The details are presented in Section 5.

It should be noted that, usually, the reconstruction algorithm $R^D(\cdot)$ takes random bits in addition to an advice string. For our applications, it is crucial that a reconstruction algorithm is *deterministic*, or, in other words, random bits must be counted as the length of advice. In contrast, for the purpose of constructing a hitting set generator based on a hard function, we do not need to count the number of random bits as advice complexity, since random bits can be fixed as a non-uniform advice of a circuit (as in, e.g., [Uma09]). Consequently, we could not find any previous black-box hitting set generator whose reconstruction procedure uses $O(m)$ random bits and an advice string of length $O(m)$ for $m = O(\log n)$ in the literature (e.g., [SU05, ISW06, TUZ07, Uma09]).

Nonetheless, we were able to construct the hitting set generator of Theorem 4.3 by combining the ideas of Umans [Uma09] and Ta-Shma, Umans, and Zuckerman [TUZ07]. Common to the ideas of [Uma09, TUZ07] is to iteratively compose a black-box construction with itself. Based on iterated compositions, Umans [Uma09] presented an "optimal" hitting set generator construction (which is not enough for our purpose because of the randomness complexity of the reconstruction algorithm). Ta-Shma, Umans, and Zuckerman [TUZ07] constructed an extractor, which is a weaker object than a black-box pseudorandom generator construction, as shown in the insightful work of Trevisan [Tre01].

The parameters achieved in [TUZ07] would have been enough for our applications. Unfortunately, they only constructed an extractor, which is an information-theoretic object and does not yield a black-box hitting set generator construction. Our main technical contribution is to make the composition of [TUZ07] work while maintaining the property of a hitting set generator.

[19] For a function $f\colon \{0,1\}^n \to \{0,1\}$, $D$ avoids $H\colon \{0,1\}^{2^n} \times \{0,1\}^d \to \{0,1\}^m$ of Theorem 4.3 in time $2^{O(n)}$, where $m = O(n)$.

Roughly speaking, the extractor Ext of [TUZ07] is constructed as follows. [TUZ07] first constructs a *lossless condenser* $C_n$ that takes an input $x$ of length $n$, and compresses $x$ to a string of length $n^{1/2}m$ while maintaining the min-entropy of the input distribution and using $O(\log n)$ random bits. Composing the condenser $C_n$ with itself $K := \log(\log n/\log m)$ times, the condenser $C' := C_{m^{O(1)}} \circ \cdots \circ C_{n^{1/4}m^{1+1/2}} \circ C_{n^{1/2}m} \circ C_n$ that compresses an $n$-bit string to an $m^{O(1)}$-bit string is constructed. In the last step, Ext is constructed by composing the condenser $C'$ with an extractor that extracts $m$ random bits from the $m^{O(1)}$-bit strings. The key idea behind the composition is that the randomness used by $C'$ is at most $\sum_{k \le K} O(2^{-k} \log n + \log m) = O(\log n)$ bits, which is better than directly constructing a condenser that compresses $n$-bit strings to $m^{O(1)}$-bit strings.

We construct our hitting set generator in a similar approach. In our case, however, it is essential that we do not apply Yao's distinguisher-to-next-bit-predictor transformation (which costs $m$ random bits to the reconstruction procedure) for each composition step. Since there are $\omega(1)$ rounds of compositions, we cannot spend $O(\log n)$ random bits per round. We overcome this difficulty as follows. First, using randomness efficient sampling (specifically, a hitter that uses $m + O(\log n)$ random bits [Gol11]), we repeat Yao's distinguisher-to-next-bit-predictor transformation so that the transformation succeeds with high probability. Then, at each composition step, we *reuse* the randomness of the reconstruction procedure, thereby spending at most $m + O(\log n)$ random bits in the overall steps. The randomness of the reconstruction procedure can be reused because we can apply a union bound to upper-bound a failure probability. The main difficulty in the proof is to find a *right* definition of a black-box hitting set generator construction to enable the composition with a small overhead. The definition can be found in Definition 4.1, and the details of the proof are presented in Section 4.

We briefly mention the relationship between our proof techniques and previous works on meta-complexity. It is common to use a black-box construction of pseudorandom generators in order to analyze meta-complexity, as first systematically explored by the work of Allender, Buhrman, Koucký, van Melkebeek, and Ronneburger [ABK+06b] (as well as subsequent works such as [CIKK16, OS17, HS17]). However, a typical construction of pseudorandom generators requires a significant amount of non-uniformity or randomness in the reconstruction procedure, which makes hardness reductions highly non-uniform or randomized. What is crucial for this work and the recent work [Hir20b] is to minimize the amount of non-uniformity, which enables us to show hardness results under *uniform and deterministic* reductions.

## 2.2 DistPH-Hardness of GapMINKT$^{\mathsf{PH}}$

We explain the proof ideas of DistPH-hardness of GapMINKT$^{\mathsf{PH}}$.

**Theorem 1.16.** *If* GapMINKT$^{\mathsf{PH}} \in \mathsf{P}$, *then* $\mathsf{DistPH} \subseteq \mathsf{AvgP}$.

This result improves and *significantly* generalizes the result of our previous work [Hir20b] that shows that GapMINKT$^{\mathsf{NP}} \in \mathsf{BPP}$ implies $\mathsf{DistNP} \subseteq \mathsf{HeurBPP}$. Specifically, our result is a hardness result under the *errorless* notion (HeurBPP versus AvgBPP or AvgP). Furthermore, we generalize DistNP-hardness to DistPH-hardness.

One might feel that it should be easy to extend DistNP-hardness of GapMINKT$^{\mathsf{NP}}$ to DistPH-hardness of GapMINKT$^{\mathsf{PH}}$. However, we emphasize that these hardness results are *fundamentally different* from the perspective of black-box reductions, and we need to overcome the "$\mathsf{S}_2^{\mathrm{p}}$-*barrier*": briefly, the DistNP-hardness can be regarded as an average-case-to-average-case reduction,

whereas the DistPH-hardness is an average-case-to-worst-case reduction and cannot be regarded as an average-case-to-average-case reduction.

First, as a special case, observe that we need to reduce any tally language $L \subseteq \{1\}^*$ in PH to GapMINKT$^{\mathsf{PH}}$. (Indeed, any sparse language must be solved exactly by considering a samplable distribution that is supported on a sparse language, as in [BCGL92].) By a padding argument, whether one can reduce a tally language in PH to GapMINKT$^{\mathsf{PH}}$ is equivalent to whether the linear-exponential-time hierarchy EH is reducible to GapMINKT$^{\mathsf{PH}}$ via a linear-exponential-time reduction (i.e., E-reduction).

With this view in mind, the starting point is the unexpected hardness results of [Hir20c] that refuted Allender's conjecture [All12]. Specifically, it was shown that $\mathsf{EXPH} \subseteq \mathsf{EXP}_{\|}^{R_{\mathrm{K}}}$, where $R_{\mathrm{K}}$ is the set of (resource-unbounded) Kolmogorov-random strings and $\mathsf{EXP}_{\|}$ stands for nonadaptive exponential-time reductions, and that $\mathsf{S}_2^{\mathrm{exp}}$ is *exactly characterized* by $\bigcap_D \mathsf{EXP}_{\|}^{D}$, where the intersection is taken over all one-sided-error heuristic oracles $D$ for $R_{\mathrm{K}}$. (More formally, $D$ is an arbitrary dense subset of $R_{\mathrm{K}}$; however, it is useful to consider $D$ as a one-sided-error heuristic algorithm for $R_{\mathrm{K}}$, which are essentially equivalent.)

By a padding argument, the proofs of [Hir20c] can be translated to a *quasi-polynomial-time* reduction from any tally language in PH to $R_{\mathrm{K}}$. This is not efficient enough. However, it was shown in [Hir20b] that the running time can be optimized using a "$k$-wise direct product generator" $\mathrm{DP}_k$, which is a black-box pseudorandom generator construction that achieves nearly optimal advice complexity.

For any $n, k \in \mathbb{N}$, the *$k$-wise direct product generator* $\mathrm{DP}_k \colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{d+k}$ is defined as $\mathrm{DP}_k(x, \bar{z}) = (z_1, \cdots, z_k, \widehat{x}(z_1), \cdots, \widehat{x}(z_k))$ for $x \in \{0,1\}^n$ and $\bar{z} := (z_1, \cdots, z_k) \in (\{0,1\}^{O(\log n)})^k$, where $\widehat{x}$ denotes an error-corrected version of $x$ and $\widehat{x}(z_i)$ denotes the $z_i$th bit of $\widehat{x}$. This is essentially a degenerated case of the Nisan–Wigderson pseudorandom generator [NW94],[20] and thus the security of $\mathrm{DP}_k$ can be easily proved using a standard hybrid argument. The important property of the simple pseudorandom generator construction $\mathrm{DP}_k$ is that it achieves the *nearly optimal* advice complexity of $k + O(\log k)$. Indeed, Trevisan and Vadhan [TV07] showed that the advice complexity of any black-box pseudorandom generator construction must be at least $(d+k) - d - O(1) = k - O(1)$, using its connection to an extractor. (Note that, unlike the advice complexity considered in Section 2.1.4, we do not count as advice complexity the number of random bits used by a reconstruction procedure.) The construction $\mathrm{DP}_k$ was a principal tool for improving the approximation quality of non-black-box worst-case to average-case reductions, as in [Hir20a] (i.e., Theorem 2.3), and will be used again in Section 2.3.

### 2.2.1 An Exposition of DistNP-Hardness

Using the fact that the advice complexity of $\mathrm{DP}_k$ is small, we review the DistNP-hardness result of GapMINKT$^{\mathsf{NP}}$ shown in [Hir20b]. For simplicity, we assume that the oracle is MINKT$^{\mathsf{NP}}$, i.e., the exact version of GapMINKT$^{\mathsf{NP}}$. Without loss of generality, it suffices to show the hardness result under the uniform distribution, owing to a result of Impagliazzo and Levin [IL90]. Consider a random input $x \sim \{0,1\}^n$. Let $V(x, y)$ be an NP verifier that takes nondeterministic bits $y$, and we aim to simulate $V$ on average. The important property of a random input $x$ is that the computational depth $\mathrm{cd}^t(x)$ of $x$ is small. Indeed, $\mathrm{cd}^t(x) := \mathrm{K}^t(x) - \mathrm{K}(x) \approx n - n = 0$ holds with

[20] The standard definition of the Nisan–Wigderson generator does not include the seed in the output; however, the generator remains secure even if the seed is included in the output [KvMS12].

high probability for any time bound $t$.

We present an algorithm that simulates $V(x,y)$ in time $2^{\mathrm{cd}^t(x)+O(\log n)}$ for some $t := \mathsf{poly}(n)$ and for every $x$ of length $n$, as expected from the work of Antunes and Fortnow [AF09] (cf. Section 1.4.2).[21] Let $y_x$ denote the lexicographically first $\mathsf{NP}$ certificate such that $V(x,y_x)=1$. Consider the candidate pseudorandom generator $\mathrm{DP}_k(y_x,\text{-})$ instantiated with $y_x$ as a candidate hard function, where $k$ is a parameter chosen later. Since $y_x$ can be computed in polynomial time with $\mathsf{NP}$ oracle, for any $\bar{z}\in\{0,1\}^d$ and for most $w\sim\{0,1\}^{d+k}$, for a sufficiently large polynomial $t=\mathsf{poly}(n)$, it holds that[22]

$$\begin{aligned}\mathrm{K}^{2t,\mathsf{NP}}(x,\mathrm{DP}_k(y_x,\bar{z}))&\le \mathrm{K}^{t,\mathsf{NP}}(x)+d+O(\log n),\\ \mathrm{K}(x,w)&\ge \mathrm{K}(x)+d+k-O(\log n),\end{aligned}$$

where the second inequality follows from the symmetry of information of (resource-unbounded) Kolmogorov complexity. By choosing $k := \mathrm{K}^{t,\mathsf{NP}}(x)-\mathrm{K}(x)+O(\log n)=\mathrm{cd}^{t,\mathsf{NP}}(x)+O(\log n)$ and defining the function $D$ as $D(w):=0$ iff $\mathrm{K}^{2t,\mathsf{NP}}(x,w)\le \mathrm{K}^{t,\mathsf{NP}}(x)+d+O(\log n)$, we obtain that $D$ is a distinguisher for $\mathrm{DP}_k(y_x,\text{-})$. Therefore, by the reconstruction property of $\mathrm{DP}_k(y_x,\text{-})$, there exists an advice string of length roughly $k$ that enables the reconstruction of $y_x$. Exhaustively trying all the advice strings in time $2^{k+O(\log n)}\le 2^{\mathrm{cd}^t(x)+O(\log n)}$, one can find the certificate $y_x$, whose correctness can be verified using $V$. (In fact, the running time is faster, and it runs in time $2^{\mathrm{cd}^{t,\mathsf{NP}}(x)+O(\log n)}$.)

### 2.2.2 $\mathsf{S}_2^{\mathrm{p}}$-Barrier

A couple of remarks on the $\mathsf{DistNP}$-hardness are in order. Firstly, it is not difficult to make the heuristic algorithm an *errorless* randomized algorithm. Indeed, one can verify that $D$ is a distinguisher by randomly sampling $w$ and verifying that $D(w)=1$ holds with high probability, in which case the algorithm can enumerate the first certificate $y_x$ for sure. Therefore, $\mathsf{DistNP}\subseteq\mathsf{AvgBPP}^D$ holds for any oracle $D$ that solves $\mathrm{GapMINKT}^{\mathsf{NP}}$. Secondly, the randomized algorithm $\mathsf{BPP}$ can be derandomized using the pseudorandom generator constructed in Section 2.1; therefore, $\mathsf{DistNP}\subseteq\mathsf{AvgBPP}=\mathsf{AvgP}$ if $\mathrm{GapMINKT}^{\mathsf{NP}}\in\mathsf{P}$.[23]

Lastly and most importantly, one can observe that the reduction to $\mathrm{GapMINKT}^{\mathsf{NP}}$ actually works even for any (black-box) one-sided-error heuristic algorithm $D$ that solves $\mathsf{coGapMINKT}^{\mathsf{NP}}$ or $R_{\mathrm{K}}$ on average. Using the characterization that $\mathsf{S}_2^{\mathrm{exp}}=\bigcap_D \mathsf{EXP}_{\|}^D$ [Hir20c], it is not hard to observe that the reduction of Section 2.2.1 can be simulated by an $\mathsf{S}_2^{\mathrm{p}}$-type computation on a unary input.[24] Here, $\mathsf{S}_2^{\mathrm{p}}$ denotes the second level of the symmetric hierarchy, and $\mathsf{P}^{\mathsf{NP}}\subseteq\mathsf{S}_2^{\mathrm{p}}\subseteq\mathsf{ZPP}^{\mathsf{NP}}$ [Can96, RS98, Cai07]. In order to extend the $\mathsf{DistNP}$-hardness to $\mathsf{DistPH}$-hardness, we must develop a fundamentally different reduction that overcomes the $\mathsf{S}_2^{\mathrm{p}}$-barrier of black-box reductions.

[21] We use the notion of computational depth for illustration purposes only, and do not use it in the actual proof.

[22] $\mathrm{K}^{t,\mathsf{NP}}(x)$ is an informal notation that should be regarded as $\mathrm{K}^{t,\mathsf{SAT}}(x)$.

[23] In fact, the number of random bits used by the reduction is at most $\mathsf{polylog}(n)$. Thus, the reduction can be regarded as a deterministic quasi-polynomial-time reduction without using the pseudorandom generator.

[24] Indeed, two competing provers send candidate one-sided-error heuristic algorithms $D_1, D_2$, respectively (by enumerating all the input-and-answer pairs), a verifier checks if $D_1$ and $D_2$ accept most inputs, and then runs the reduction with oracle $D_1\wedge D_2$, which is guaranteed to be a one-sided-error algorithm. Since the reduction of Section 2.2.1 can be regarded as a deterministic quasi-polynomial-time reduction, the running time of the $\mathsf{S}_2^{\mathrm{p}}$-type simulation is quasi-polynomial.

### 2.2.3 DistPH-Hardness

Now we explain the idea of the reduction that proves the DistPH-hardness of GapMINKT$^{\mathsf{PH}}$ and overcomes the $\mathsf{S}_2^{\mathrm{p}}$-barrier. For simplicity, we present Dist$\mathbf{\Sigma}_2^{\mathrm{p}}$-hardness of MINKT$^{\mathsf{PH}}$. Let $L \in \mathbf{\Sigma}_2^{\mathrm{p}}$ and $V(x, y, z)$ be a $\mathbf{\Sigma}_2^{\mathrm{p}}$-type verifier such that $x \in L$ if and only if $\exists y, \forall z, V(x, y, z) = 1$. Our goal is to simulate $V$ on average.

A basic proof idea is reminiscent of the result of [Hir20c] showing that $\mathsf{EXPH} \subseteq \mathsf{EXP}^{R_{\mathrm{K}}}_{\|}$. Specifically, we take small lists $\mathcal{L}_x, \mathcal{L}_{x,y}$ such that $x \in L$ if and only if $\exists y \in \mathcal{L}_x, \forall z \in \mathcal{L}_{x,y}, V(x, y, z) = 1$.[25] Note that, if the lists are small and efficiently computable, we can easily simulate $V$ by trying every $y \in \mathcal{L}_x$ and $z \in \mathcal{L}_{x,y}$. The question is how to define these lists.

**A First Attempt.** Following the idea of DistNP-hardness of Section 2.2.1, the first attempt would be to define the list $\mathcal{L}_x$ as the output of all the possible outputs of the reconstruction procedure given the distinguisher $D$ as oracle. Note that the size of the list is approximately at most $|\mathcal{L}_x| \lesssim 2^{\mathrm{cd}^t(x)}$. We can define the list $\mathcal{L}_{x,y}$ in the exactly same way so that $|\mathcal{L}_{x,y}| \lesssim 2^{\mathrm{cd}^t(x,y)}$. Unfortunately, this approach is problematic. There may exist $y \in \mathcal{L}_x$ such that $\mathrm{cd}^t(x, y) \geq |y|$, which is significantly large; as a consequence, the list $\mathcal{L}_{x,y}$ becomes too large to be enumerated efficiently.

**Our Solution.** We resolve this issue as follows. We define a partial list $\mathcal{L}'_x$ of $\mathcal{L}_x$ as $\{\, y \in \mathcal{L}_x \mid \mathrm{K}^{t,\mathsf{PH}}(x, y) \leq |x| + O(\log n) \,\}$. The key idea behind this definition is that we do not have to consider every candidate $y$. Recall that the goal is to simulate $\exists y, \forall z, V(x, y, z) = 1$, and thus it suffices to find the lexicographically first certificate $y_x$. In particular, we have $\mathrm{K}^{t,\mathsf{PH}}(x, y) \leq |x| + O(\log n)$ for some large $t$ and $y := y_x$. We check this condition, and we take the partial list $\mathcal{L}'_x$ of the certificates $y$ that passed the test. We then define $\mathcal{L}_{x,y}$ as before, except that we only consider $y \in \mathcal{L}'_x$. The correctness of this algorithm can be proved easily (i.e., $x \in L$ if and only if $\exists y \in \mathcal{L}'_x, \forall z \in \mathcal{L}_{x,y}, V(x, y, z) = 1$).

In order to bound the running time of the reduction, we need to claim that the size $|\mathcal{L}_{x,y}|$ of the list $\mathcal{L}_{x,y}$ is not large for any $y \in \mathcal{L}'_x$. This can be shown for a random input $x \sim \{0,1\}^n$ as follows. Recall that $\mathrm{K}(x) \approx n$ holds with high probability by a simple counting argument. Therefore, $\mathrm{cd}^{t,\mathsf{PH}}(x, y) = \mathrm{K}^{t,\mathsf{PH}}(x, y) - \mathrm{K}(x, y) \lesssim n + O(\log n) - n = O(\log n)$, from which it follows that $|\mathcal{L}_{x,y}| \lesssim 2^{\mathrm{cd}^{t,\mathsf{PH}}(x,y)} = n^{O(1)}$ as desired.

**$\mathsf{S}_2^{\mathrm{p}}$-Barrier** Finally, let us see why our DistPH-hardness reduction overcomes the $\mathsf{S}_2^{\mathrm{p}}$-barrier discussed earlier. Specifically, we explain the reason why the reduction is not a "black-box" reduction that works for *any* one-sided-error heuristic algorithm. The reason is that the partial list $\mathcal{L}'_x$ cannot be defined with a one-sided-error heuristic oracle that solves coMINKT$^{\mathsf{PH}}$. Indeed, what is crucial in the reduction above is to reject *every* $y$ such that $\mathrm{K}^{t,\mathsf{PH}}(x, y) > |x| + O(\log n)$. Although this is possible with a MINKT$^{\mathsf{PH}}$ oracle, it is not necessarily possible with a one-sided-error heuristic oracle for coMINKT$^{\mathsf{PH}}$.

## 2.3 Error-Tolerant Worst-Case-To-Average-Case Reduction

The proof techniques (and the previous result of Theorem 2.3 [Hir20a]) presented so far are sufficient to show the equivalence between DistPH $\subseteq$ AvgP and GapMINKT$^{\mathsf{PH}} \in \mathsf{P}$. However, they

[25] In [Hir20c], the lists $\mathcal{L}_x$ and $\mathcal{L}_{x,y}$ are defined as all the truth tables of circuits of size $p(|x|)$ and $q(|x|)$, respectively, where $p$ and $q$ are polynomials such that $p \ll q$.

are insufficient for obtaining the hardness amplification theorem for $\mathsf{PH}$. We now explain the error-tolerant non-black-box worst-case-to-average-case reduction, which brings us from the average-case complexity world to the worst-case complexity world.

**Theorem 1.15.** *Let $c > 0$ be any constant and $A$ be any $\mathsf{NP}$-hard oracle. If $\{\mathsf{coMINKT}^A\} \times \mathsf{PSamp} \subseteq \mathsf{Avg}^1_{1-n^{-c}}\mathsf{P}$, then $\mathrm{Gap}(\mathrm{K}^A \text{ vs } \mathrm{K}) \in \mathsf{P}$.*

First, the pseudorandom generator constructed in Section 2.1 (under the assumption that $\mathrm{GapMINKT}^{\mathsf{NP}} \in \mathsf{P}$) can be obtained here as well. Thus, in the remainder of the proof sketch, we make use of a nearly optimal pseudorandom generator $G = \{G_n : \{0,1\}^{O(\log n)} \to \{0,1\}^n\}_{n\in\mathbb{N}}$.

As explained in Section 1.3, the improvement is due to the fact that the time bound $t$ of MINKT can actually be fixed to $n^\gamma$, where $\gamma > 0$ is an arbitrary constant. Consider a version of MINKT whose parameters are fixed as follows.

**Definition 2.6.** *For functions $t, s\colon \mathbb{N} \to \mathbb{N}$ and an oracle $A$, let $\mathrm{MINKT}^A[t = t(n), s = s(n)]$ denote the language $\{\, x \in \{0,1\}^* \mid \mathrm{K}^{t(|x|),A}(x) \le s(|x|) \,\}$.*

Then, we show a non-black-box worst-case-to-average-case reduction from $\mathrm{Gap}(\mathrm{K}^A \text{ vs } \mathrm{K})$ to the distributional problem $(\mathsf{coMINKT}^A[t = n^\gamma, s = n - \log(n/\delta(n))], \mathcal{U})$, where $\mathcal{U} = \{\mathcal{U}_n\}_{n\in\mathbb{N}}$ denotes the family of the uniform distributions $\mathcal{U}_n$ over $\{0,1\}^n$.

**Lemma 8.9.** *Let $\gamma > 0$ be any constant, $\delta\colon \mathbb{N} \to (0,1)$ be any function such that $\delta(n)^{-1} = n^{O(1)}$, and $A$ be any oracle. Assume that $(\mathsf{coMINKT}^A[t = n^\gamma, s = n - \log(n/\delta(n))], \mathcal{U}) \in \mathsf{Avg}^1_{1-\delta(n)}\mathsf{P}$ and that there exists a nearly optimal pseudorandom generator. Then, $\mathrm{Gap}(\mathrm{K}^A \text{ vs } \mathrm{K}) \in \mathsf{P}$.*

We emphasize that the existence of a pseudorandom generator in the assumption of Lemma 8.9 is essential for the proof; without the existence of a pseudorandom generator, we do not know how to prove the correctness of a randomized reduction. In fact, we make use of the pseudorandom generator *thrice*—once for derandomizing the randomized non-black-box reductions, once for derandomizing a reconstruction algorithm (which enables us to improve the approximation quality of $\mathrm{Gap}(\mathrm{K}^A \text{ vs } \mathrm{K})$), and once for making the time bound $t$ large.

We review the proof idea of Theorem 2.3 [Hir20a]. The key is to use the $k$-wise direct product generator $\mathrm{DP}_k\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{d+k}$ that achieves the nearly optimal advice complexity. Let $(x, 1^t, 1^s)$ be an input to $\mathrm{Gap}(\mathrm{K}^A \text{ vs } \mathrm{K})$. The reduction is a simple randomized reduction that maps the instance $(x, 1^t, 1^s)$ to an instance $(\mathrm{DP}_k(x,z), 1^{t'}, 1^{d+k-2})$ of $\mathsf{coMINKT}^A$, where $n := |x|$, $k := s + O(\log n)$, $t' = \mathsf{poly}(n,t)$, and $z \sim \{0,1\}^d$.

The correctness of the reduction can be proved using the reconstruction property of $\mathrm{DP}_k$ and the existence of a pseudorandom generator. Roughly speaking, the idea of the reduction is that the instance $x$ of length $n$ is mapped to a *compressed* instance $\mathrm{DP}_k(x,z)$ of length $d+s+O(\log n)$. It is easy to see that any instance $x$ that can be compressed to $s$ bits is mapped to an instance $\mathrm{DP}_k(x,z)$ that can be compressed. Conversely, if $x$ cannot be compressed, $\mathrm{DP}_k(x,z)$ cannot be distinguished from the uniform distribution by the one-sided-error heuristic algorithm $\mathsf{coMINKT}^A$, which can be proved using the reconstruction property. Note here that the reconstruction procedure of $\mathrm{DP}_k$ uses a significant amount of $d = O(k \log n)$ random bits; using the nearly optimal pseudorandom generator, one can reduce the randomness, which enables us to improve the approximation error of $\mathrm{GapMINKT}^A$ from an $O(\log n)$-factor approximation to an additive error of $O(\log n)$.

Note that the time parameter $t$ is mapped to some $t'$. In order to solve $\mathrm{Gap}(\mathrm{K}^A \text{ vs } \mathrm{K})$ on an instance $(x, 1^t, 1^s)$ for *every* $t$, it was previously assumed that the heuristic algorithm for $\mathsf{coMINKT}^A$ succeeds for every $t'$, which is the reason why the required success probability was significantly large.

### 2.3.1 A Padding Argument

Now, we would like to extend the reduction of [Hir20a] described above to a reduction to $\mathsf{coMINKT}^A[t = n^\gamma, s = s(n)]$. For simplicity, we describe the case when $s(n) = n - 2$ and $\gamma = 2$.

The main idea is to use a padding argument. In fact, it is easy to see that $\mathrm{GapMINKT} \in \mathsf{P}$ if and only if $\mathrm{GapMINKT}[t = n^2] \in \mathsf{P}$: Given an instance $(x, 1^t, 1^s)$ of $\mathrm{GapMINKT}^A$, one can map it to an instance $(x1^t, 1^{s+O(\log n)})$ of $\mathrm{GapMINKT}[t = n^2]$. Since the length of the string $x1^t$ is at least $t$, $x1^t$ can be compressed in $t^2$ time steps (see Proposition 3.6 for a formal proof).

The padding argument that maps $x$ to $x1^t$ works in the *worst-case-to-worst-case* reduction. However, the same padding argument does not work in the case of worst-case-to-average-case reductions. For this, we instead pad the length of $x$ by mapping $x$ to $x \cdot w$, where $w$ is a string chosen at random from a uniform distribution.

More precisely, we map an instance $(x, 1^t, 1^s)$ of $\mathrm{GapMINKT}^A$ to an instance $\mathrm{DP}_k(x, z) \cdot w$ of $\mathsf{coMINKT}^A[t = n^2, s = n - 2]$, where $w \sim \{0,1\}^t$. The correctness of the reduction can be shown roughly as follows. If $x$ can be compressed to $s$ bits, then $\mathrm{DP}_k(x, z) \cdot w$ can be compressed to a string of approximate length $s + d + t < k + d + t - 2$; thus, $\mathrm{DP}_k(x, z) \cdot w$ is rejected by a one-sided-error heuristic algorithm $D$ for $\mathsf{coMINKT}^A$ (by choosing $k :\approx s$). Conversely, if $x$ cannot be compressed, then $D$ cannot distinguish $\mathrm{DP}_k(x, z) \cdot w$ from the uniform distribution, which can be shown by derandomizing the choice of random bits $w$ as well as $z$. A formal proof is presented in Section 8.

### 2.3.2 Practically Generating Hard NP Instances?

Finally, we note that the error-tolerant worst-case-to-average-case reduction could be used to practically generate hard NP instances. Specifically, under the plausible assumptions that there exist (cryptographic and complexity-theoretic) pseudorandom generators, a $(1-1/\mathsf{poly}(n))$-fraction of instances of MINKT require super-polynomial time to solve.

**Corollary 2.7** (of Lemma 8.9)**.** *Assume that* $\mathsf{E} \not\subseteq \bigcap_{\epsilon>0} \mathsf{i.o.SIZE}(2^{\epsilon n})$ *and* $\mathrm{GapMINKT} \notin \mathsf{P}$*. Let* $c > 0$ *be any constant. Then, for any algorithm* $M$ *that solves* $\mathrm{MINKT}[t = n^{1/10}, s = n-(c+1)\log n]$ *and any polynomial* $p$*, for infinitely many* $n \in \mathbb{N}$*, with probability at least* $1 - n^{-c}$ *over the choice of* $x \sim \{0,1\}^n$*,* $t_M(x) \geq p(n)$ *holds, where* $t_M(x)$ *denotes the running time of* $M$ *on input* $x$*.*

*Proof.* Define $L := \mathrm{MINKT}[t = n^{1/10}, s = n-(c+1)\log n]$. Assume, for sake of contradiction, that there exist a constant $c$, a polynomial $p$, and an algorithm $M$ such that $M$ solves $L$ and

$$\Pr_{x\sim\{0,1\}^n}[t_M(x) \geq p(n)] < 1 - n^{-c} \tag{1}$$

for all large $n \in \mathbb{N}$. Define an algorithm $M'$ so that $M'(x) := M(x)$ if $M$ halts on input $x \in \{0,1\}^n$ in time $p(n)$; otherwise, $M'(x) := \bot$. Then, $M'$ is an errorless heuristic algorithm for $(L, \mathcal{U})$, and its failure probability is less than $1 - n^{-c}$ by Eq. (1). Therefore, $(L, \mathcal{U}) \in \mathsf{Avg}_{1-n^{-c}}\mathsf{P} \subseteq \mathsf{Avg}^1_{1-n^{-c}}\mathsf{P}$.

The circuit lower bound assumption $\mathsf{E} \not\subseteq \bigcap_{\epsilon>0} \mathsf{i.o.SIZE}(2^{\epsilon n})$ implies the existence of a nearly optimal pseudorandom generator [IW97]. Therefore, we can apply Lemma 8.9 and conclude that $\mathrm{GapMINKT} = \mathrm{Gap(K\ vs\ K)} \in \mathsf{P}$, which is a contradiction. □

The assumption that $\mathrm{GapMINKT} \notin \mathsf{P}$ is relatively mild. For example, any one of the following assumptions implies $\mathrm{GapMINKT} \notin \mathsf{P}$: $\mathsf{SZK} \not\subseteq \mathsf{BPP}$ [AD17], the existence of cryptographic pseudorandom generators, the existence of (auxiliary-input) one-way functions [HILL99], unsolvability

of Random 3SAT in polynomial time [HS17], or more generally, the existence of a cryptographic hitting set generator [Hir18].

### 2.4 Organization and Proof of the Main Theorem

Here, we provide a proof of the main theorem and explain the organization of the remainder of the paper.

*Proof of Theorem 1.14.* (Item 1 ⇒ 4), (Item 4 ⇒ 6), (Item 6 ⇒ 7), (Item 1 ⇒ 5), (Item 5 ⇒ 7), (Item 8 ⇒ 2), and (Item 2 ⇒ 3) are obvious from the definitions.

(Item 7 ⇒ 8) follows from the error-tolerant non-black-box worst-case-to-average-case reduction (Theorem 1.15), whose proof is given in Section 8.

(Item 3 ⇒ 1) follows from the $\mathsf{DistPH}$-hardness of $\mathrm{GapMINKT}^{\mathsf{PH}}$ (Theorem 1.16), whose proof is given in Section 7.

(Item 2 ⇔ 9) follows from the simple padding argument (Proposition 3.6).

(Item 1 ⇒ 10) follows from Fact 3.7 and Theorem 6.5, the latter of which implies $\mathsf{P} = \mathsf{BPP}$. Such a derandomization statement is proved in Section 6 based on the $\mathsf{E}^{\mathsf{NP}}$-hardness of Section 5.

(Item 10 ⇒ 11) and (Item 11 ⇒ 12) are obvious.

(Item 12 ⇒ 3) follows from Theorem 8.10. □

## 3 Preliminaries

For a language $L \subseteq \{0,1\}^*$, let $\mathsf{co}L$ denote the complement of $L$, i.e., $\mathsf{co}L := \{0,1\}^* \setminus L$. Let $\mathsf{co}\mathfrak{C}$ denote $\{\, \mathsf{co}L \mid L \in \mathfrak{C} \,\}$ for a class $\mathfrak{C}$ of languages. A *promise problem* $\Pi$ is a pair $(\Pi_{\mathrm{YES}}, \Pi_{\mathrm{NO}})$ of languages $\Pi_{\mathrm{YES}}, \Pi_{\mathrm{NO}} \subseteq \{0,1\}^*$. Unlike in the standard definition, we do *not* require that $\Pi_{\mathrm{YES}} \cap \Pi_{\mathrm{NO}} = \varnothing$. A promise problem $\Pi$ is said to be *solved* by an algorithm $A$ if $A$ accepts any instance in $\Pi_{\mathrm{YES}}$ and rejects any instance in $\Pi_{\mathrm{NO}}$, from which it follows that $\Pi_{\mathrm{YES}} \cap \Pi_{\mathrm{NO}} = \varnothing$. If $\Pi_{\mathrm{YES}} = \mathsf{co}\Pi_{\mathrm{NO}}$, we identify $\Pi$ with the language $\Pi_{\mathrm{YES}}$. For a semantic class $\mathfrak{C}$, such as $\mathsf{BPP}$ and $\mathsf{MA}$, we denote its promise version by pr-$\mathfrak{C}$; we often identify $\mathfrak{C}$ with pr-$\mathfrak{C}$ for a syntactic class $\mathfrak{C}$.

### 3.1 Pseudorandomness

A family of functions $G = \{G_n : \{0,1\}^{s(n)} \to \{0,1\}^n\}_{n\in\mathbb{N}}$ is said to be a *pseudorandom generator* $\epsilon$-secure against a class $\mathfrak{C}$ if every $C \in \mathfrak{C}$ is *$\epsilon$-fooled* by $G$. Here, we say that $G$ $\epsilon$-fools a function $C\colon \{0,1\}^n \to \{0,1\}$ if $|\Pr_{z\sim\{0,1\}^{s(n)}}[C(G_n(z)) = 1] - \Pr_{w\sim\{0,1\}^n}[C(w) = 1]| < \epsilon$. A function $C\colon \{0,1\}^n \to \{0,1\}$ is said to *$\epsilon$-distinguish* $G$ (from the uniform distribution) if $G$ does not $\epsilon$-fool $C$. By default, we choose $\epsilon := 1/n$.

In this paper, we mainly consider a *complexity-theoretic* pseudorandom generator, which has more computational resources than the adversary. This notion of pseudorandom generator is not useful for cryptography, but it is sufficient for derandomizing two-sided-error randomized algorithms. Specifically, a pseudorandom generator is said to be *explicit* if there exists a deterministic algorithm that takes a seed $z \in \{0,1\}^{s(n)}$ and $n \in \mathbb{N}$ and outputs $G_n(z)$ in time $2^{O(s(n))} \cdot n^{O(1)}$ for any $n \in \mathbb{N}$. We call an explicit pseudorandom generator $G = \{G_n : \{0,1\}^{O(\log n)} \to \{0,1\}^n\}_{n\in\mathbb{N}}$ secure against linear-sized circuits a *nearly optimal pseudorandom generator*.

In contrast, we mainly consider a (cryptographic) hitting set generator that has less computational resources than the adversary. A family of functions $H = \{H_n : \{0,1\}^{s(n)} \to \{0,1\}^n\}_{n\in\mathbb{N}}$

is said to be a *hitting set generator* $\epsilon$-secure against a complexity class $\mathfrak{C}$ if no algorithm $C \in \mathfrak{C}$ *$\epsilon$-avoids* $H$. Here, we say that $C \in \mathfrak{C}$ $\epsilon$-avoids $H$ if, for all large $n \in \mathbb{N}$, $C(H_n(z)) = 0$ for every $z \in \{0,1\}^{s(n)}$ and $\Pr_{w \sim \{0,1\}^n}[C(w) = 1] \geq \epsilon$. Let $\mathrm{Im}(H_n)$ denote the image $\{ H_n(z) \mid z \in \{0,1\}^{s(n)} \}$ of $H_n$. The condition that $C$ $\epsilon$-avoids $H$ is equivalent to saying that, for all large $n \in \mathbb{N}$, $C$ rejects every $z \in \mathrm{Im}(H_n)$ and accepts an $\epsilon$-fraction of inputs. By default, we choose $\epsilon := 1/4$ and often omit the parameter $\epsilon$.

### 3.2 Average-Case Complexity

Below, we review the notions of $\mathsf{AvgP}, \mathsf{AvgBPP}$, and $\mathsf{Avg}^1\mathsf{P}$ and observe that $\mathsf{AvgBPP}$ can be derandomized under the assumption that there exists a nearly optimal pseudorandom generator.

**Definition 3.1** (Randomized Errorless Heuristic Scheme [BT06a])**.** *Let $(L, \mathcal{D})$ be a distributional problem. A randomized algorithm $A$ is said to be a* randomized errorless heuristic scheme *for $(L, \mathcal{D})$ if*

1. *$A(x, \delta, n)$ halts in time $\mathsf{poly}(n/\delta)$ for every $n \in \mathbb{N}$, $x \in \mathsf{supp}(\mathcal{D}_n)$ and $\delta^{-1} \in \mathbb{N}$,*
2. *$\Pr_A[A(x, \delta, n) \notin \{L(x), \bot\}] \leq 1/8$ for every $n \in \mathbb{N}$, $x \in \mathsf{supp}(\mathcal{D}_n)$ and $\delta^{-1} \in \mathbb{N}$, and*
3. *$\Pr_{x \sim \mathcal{D}_n}[\Pr_A[A(x, \delta, n) = \bot] \geq 1/8] \leq \delta$ for every $n, \delta^{-1} \in \mathbb{N}$.*

*The parameter $\delta$ and $1 - \delta$ are referred to as the failure and success probabilities, respectively.*

*If, in addition, the algorithm $A$ is deterministic, $A$ is said to be a* deterministic errorless heuristic scheme*.*

*The class $\mathsf{AvgP}$ (resp., $\mathsf{AvgBPP}$) is defined as the class of distributional problems for which there exists a deterministic (resp., randomized) errorless heuristic scheme. For a function $\delta \colon \mathbb{N} \to \mathbb{N}$, the class $\mathsf{Avg}_\delta\mathsf{P}$ is defined as the class of distributional problems $(L, \mathcal{D})$ for which there exists a $\mathsf{poly}(n)$-time deterministic errorless algorithm with failure probability $\delta(n)$ when inputs are sampled from $\mathcal{D}_n$ for each $n \in \mathbb{N}$.*

We consider only the family $\{\mathcal{D}_n\}_{n \in \mathbb{N}}$ of distributions such that there is an efficient algorithm that takes $x \in \mathsf{supp}(\mathcal{D}_n)$ and computes $n$; in this case, the input $n$ of $A(x, \delta, n)$ is redundant; thus, we omit the input $n$ in the rest of the paper.

We define a one-sided-error variant of heuristic algorithms as follows.

**Definition 3.2** (One-sided-error Heuristic Scheme)**.** *For a distributional problem $(L, \mathcal{D})$, a polynomial-time algorithm $A$ is said to be a* one-sided-error heuristic scheme *for $(L, \mathcal{D})$ if*

1. *$A(x, \delta)$ halts in time $\mathsf{poly}(n/\delta)$ for every $n \in \mathbb{N}, x \in \mathsf{supp}(\mathcal{D}_n)$ and $\delta^{-1} \in \mathbb{N}$,*
2. *$L(x) = 0$ implies $A(x, \delta) = 0$ for every $n \in \mathbb{N}$, $x \in \mathsf{supp}(\mathcal{D}_n)$ and $\delta^{-1} \in \mathbb{N}$, and*
3. *$\Pr_{x \sim \mathcal{D}_n}[A(x, \delta) = L(x)] \geq 1 - \delta$ for every $n, \delta^{-1} \in \mathbb{N}$.*

*For a function $\delta \colon \mathbb{N} \to (0, 1)$, an algorithm $A$ is said to be a polynomial-time one-sided-error heuristic algorithm for $(L, \mathcal{D})$ with failure probability $\delta$ if*

1. *$A(x)$ halts in time $\mathsf{poly}(n)$ for every $n \in \mathbb{N}$ and $x \in \mathsf{supp}(\mathcal{D}_n)$,*
2. *$L(x) = 0$ implies $A(x) = 0$ for every $n \in \mathbb{N}$ and $x \in \mathsf{supp}(\mathcal{D}_n)$, and*

3. $\Pr_{x\sim\mathcal{D}_n}[A(x) = L(x)] \geq 1 - \delta(n)$ *for every* $n \in \mathbb{N}$.

$\mathsf{Avg}^1\mathsf{P}$ *is defined as the class of distributional problems for which there exists a one-sided-error heuristic scheme. For a function* $\delta\colon \mathbb{N} \to (0,1)$, $\mathsf{Avg}^1_\delta\mathsf{P}$ *is defined as the class of distributional problems for which there exists a polynomial-time one-sided-error heuristic scheme with failure probability* $\delta$.

It is easy to observe that the existence of an errorless heuristic algorithm is stronger than that of a one-sided-error heuristic algorithm.

**Fact 3.3.** $\mathsf{AvgP} \subseteq \mathsf{Avg}^1\mathsf{P}$ *and* $\mathsf{Avg}_\delta\mathsf{P} \subseteq \mathsf{Avg}^1_\delta\mathsf{P}$ *for every* $\delta\colon \mathbb{N} \to (0,1)$.

*Proof Sketch.* Given an errorless heuristic algorithm $A$, a one-sided-error heuristic algorithm $A'$ can be defined as $A'(x) = 1$ if $A(x) = 1$ and $A'(x) = 0$ if $A(x) \in \{0, \bot\}$. □

**Proposition 3.4.** *Suppose that there exists an explicit pseudorandom generator* $G := \{G_n : \{0,1\}^{O(\log n)} \to \{0,1\}^n\}_{n\in\mathbb{N}}$ *secure against linear-sized circuits. Then,* $\mathsf{AvgP} = \mathsf{AvgBPP}$.

*Proof.* Let $(L, \mathcal{D}) \in \mathsf{AvgBPP}$ and let $A$ be a randomized errorless heuristic scheme for $(L, \mathcal{D})$ that uses $m := \mathsf{poly}(|x|/\delta)$ random bits on input $(x, \delta)$. Define $B$ as a deterministic algorithm that, on input $(x, \delta)$, outputs the majority of $A(x, \delta; G_m(z)) \in \{0, 1, \bot\}$ over the choice of $z \in \{0,1\}^{O(\log(|x|/\delta))}$ (here, $A(x, \delta; w)$ denotes the output of $A$ on input $(x, \delta)$ and random bits $w \in \{0,1\}^m$). Since $G_m$ is explicit, $B$ can be implemented as an algorithm that runs in time $\mathsf{poly}(|x|/\delta)$.

We claim below that $B$ is a deterministic errorless heuristic scheme for $(L, \mathcal{D})$. First, we show that $B$ does not make any error. Indeed, by Item 2 of Definition 3.1,

$$\Pr_z[A(x, \delta; G_m(z)) \in \{L(x), \bot\}] \geq \Pr_w[A(x, \delta; w) \in \{L(x), \bot\}] - o(1) \geq 7/8 - o(1) > 2/3,$$

which means that the incorrect answer $1-L(x)$ cannot be the majority of $A(x, \delta; G_m(z)) \in \{0, 1, \bot\}$. Second, we show that $\Pr_{x\sim\mathcal{D}_n}[B(x, \delta) = \bot] \leq \delta$ for every $n, \delta^{-1} \in \mathbb{N}$. Indeed, by Item 3 of Definition 3.1, with probability at least $1-\delta$ over the choice of $x \sim \mathcal{D}_n$, it holds that $\Pr_w[A(x, \delta; w) = \bot] < 1/8$. By the security of the pseudorandom generator,

$$\Pr_z[A(x, \delta; G_m(z)) = \bot] \leq \Pr_w[A(x, \delta; w) = \bot] + o(1) \leq 1/8 + o(1) < 1/3,$$

from which it follows that $\bot$ is not the majority of $A(x, \delta; G_m(z)) \in \{0, 1, \bot\}$. □

### 3.3 Basic Facts

Here, we prove some basic facts. The following is the fundamental principle of Kolmogorov-randomness. (A string $x$ is said to be *Kolmogorov-random* if $\mathrm{K}(x) \geq s(|x|)$, where $s\colon \mathbb{N} \to \mathbb{N}$ is some threshold, such as $s(n) := n - 1$.)

**Fact 3.5** (standard counting argument)**.** *For any* $s \geq 1$*, the number of strings* $x \in \{0,1\}^*$ *such that* $\mathrm{K}(x) < s$ *is less than* $2^s$.

*Proof.* The number of programs of length less than $s$ is at most $\sum_{i=0}^{s-1} 2^i < 2^s$. □

A simple padding argument shows the equivalence between sublinear-time-bounded Kolmogorov complexity and polynomial-time-bounded Kolmogorov complexity:

**Proposition 3.6** (Item 2 $\Leftrightarrow$ 9 of Theorem 1.14)**.** *For any oracle $A$ and any constant $\gamma > 0$,* $\mathrm{GapMINKT}^A \in \mathsf{P}$ *if and only if* $\mathrm{GapMINKT}^A[t = n^\gamma] \in \mathsf{P}$

*Proof.* It is obvious that $\mathrm{GapMINKT}^A[t = n^\gamma]$ is reducible to $\mathrm{GapMINKT}^A$. Conversely, let $\mathrm{Gap}_\tau\mathrm{MINKT}^A \in \mathsf{P}$ for some polynomial. Let $(x, 1^t, 1^s)$ be an instance of $\mathrm{GapMINKT}^A$, and map it to $(x1^m, 1^{s+O(\log n)})$ for $m := (t + O(\log n))^{1/\gamma}$, where $n := |x|$. We claim that this is a many-one reduction to $\mathrm{Gap}_\tau\mathrm{MINKT}^A$. Since

$$\mathrm{K}^{(n+m)^\gamma,A}(x1^m) \leq \mathrm{K}^{t+O(\log n),A}(x1^m) \leq \mathrm{K}^{t,A}(x) + O(\log n),$$

any YES instance of $\mathrm{GapMINKT}^A$ is mapped to a YES instance of $\mathrm{Gap}_\tau\mathrm{MINKT}^A$. Similarly,

$$\mathrm{K}^{\tau'(n,t),A}(x) \leq \mathrm{K}^{\tau(n+m,t)+O(\log n),A}(x) \leq \mathrm{K}^{\tau(n+m,t),A}(x1^m) + O(\log n)$$

holds for a large polynomial $\tau'$. Therefore, if $\mathrm{K}^{\tau'(n,t),A}(x) > s + \log \tau'(n,t)$, then $\mathrm{K}^{\tau(n+m,t),A}(x1^m) + O(\log n) - \log \tau'(n,t) > s$. By choosing sufficiently large $\tau'$, we obtain $\mathrm{K}^{\tau(n+m,t),A}(x1^m) + \log \tau(n+m,t) > s + O(\log n)$, which means that $(x1^m, 1^{s+O(\log n)})$ is a NO instance of $\mathrm{Gap}_\tau\mathrm{MINKT}^A$. □

We observe that the existence of a $\mathsf{P}^A$-computable hitting set generator implies average-case hardness of $\mathsf{DistNP}^A$.

**Fact 3.7.** *Let $A$ be any oracle. If $\mathsf{DistNP}^A \subseteq \mathsf{Avg}_{1/4}\mathsf{P}$, then no $\mathsf{P}^A$-computable hitting set generator $H = \{H_n : \{0,1\}^{n-1} \to \{0,1\}^n\}_{n\in\mathbb{N}}$ is $(1/4)$-secure against $\mathsf{P}$.*

*Proof.* Consider $L := \mathrm{Im}(H) = \{\, x \in \{0,1\}^* \mid x = H_n(z) \text{ for some } z \in \{0,1\}^{n-1} \,\}$. Since $H$ is computable in polynomial time with oracle $A$, the language $L$ is in $\mathsf{NP}^A$. Therefore, $(L, \mathcal{U}) \in \mathsf{DistNP}^A \subseteq \mathsf{Avg}_{1/4}\mathsf{P}$. Let $M$ be a polynomial-time errorless heuristic algorithm for $(L, \mathcal{U})$ with failure probability $1/4$. Define a polynomial-time machine $M'$ such that $M'(x) := 0$ if $M(x) \in \{1, \perp\}$, and $M'(x) := 1$ if $M(x) = 0$ for each $x \in \{0,1\}^*$.

We claim that $M'$ $(1/4)$-avoids $H$. For every $n \in \mathbb{N}$ and $z \in \{0,1\}^{n-1}$, $H_n(z) \in L$; thus $M(H_n(z)) \neq 0$ and $H_n(z)$ is rejected by $M'$. Moreover,

$$\begin{aligned}
&\Pr_{w\sim\{0,1\}^n}\left[M'(w) = 1\right] \\
&= \Pr_{w\sim\{0,1\}^n}\left[M(w) \neq \perp\right] - \Pr_{w\sim\{0,1\}^n}\left[M(w) = 1\right] \\
&\geq \frac{3}{4} - \frac{1}{2} = \frac{1}{4},
\end{aligned}$$

where the last inequality uses the fact that $\Pr_w[M(w) = 1] \leq \Pr_w[w \in L] \leq 1/2$. □

# 4 Improved Black-Box Hitting Set Generator Construction

In this section, we construct a black-box hitting set generator whose reconstruction algorithm uses few random bits and seed length and advice complexity are small. More specifically, in the following definition, we aim to minimize the randomness complexity $r$, seed length $d$, advice complexity $a$, and list complexity $L$.

**Definition 4.1** (Black-box Hitting Set Generator Construction; BB-HSG)**.** *A triple* $(H, A, R)$*, where*

- $H\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ *is called a* HSG function*,*
- $A\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^a$ *is called an* advice function*, and*
- $R^D\colon \{0,1\}^a \times \{0,1\}^d \times \{0,1\}^r \to (\{0,1\}^n)^L$ *is called a* reconstruction procedure*, it takes an oracle* $D\colon \{0,1\}^m \to \{0,1\}$*, and its output* $(\{0,1\}^n)^L$ *is regarded as the set of* $L$ *n-bit strings,*

*is called a* black-box hitting set generator construction (BB-HSG) *with advantage* $\epsilon$ *and error probability* $\delta$ *if, for any* $x \in \{0,1\}^n$ *and any* $D\colon \{0,1\}^m \to \{0,1\}$ *that* $\epsilon$*-avoids* $H(x,\text{-})\colon \{0,1\}^d \to \{0,1\}^m$*, it holds that*

$$\Pr_{w\sim\{0,1\}^r}\left[\exists z \in \{0,1\}^d,\ x \in R^D(A(x,z),z,w)\right] \geq 1-\delta.$$

*The parameters are called as follows:* $a$ *is the advice complexity,* $d$ *is the seed length,* $r$ *is the randomness complexity, and* $L$ *is the list complexity.*

*We say that a BB-HSG* $(H, A, R^{(\text{-})})$ *is* explicit *if there exists a* $\mathsf{poly}(n)$*-time algorithm for computing* $H$ *and* $A$ *and a* $\mathsf{poly}(n)$*-time* $D$*-oracle algorithm for computing* $R^D$ *for any oracle* $D$*. We implicitly assume that a BB-HSG is always explicit.*

It is important to minimize $2^{a+d+r} \cdot L$ for the following reason.

**Proposition 4.2.** *Let* $(H\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m, A, R)$ *be a BB-HSG. Then, for every function* $D\colon \{0,1\}^m \to \{0,1\}$*, there exists a list* $\mathcal{L}_D$ *of* $n$*-bit strings such that, for any* $x \in \{0,1\}^n$ *such that* $D$ $\epsilon$*-avoids* $H(x,\text{-})$*, the list* $\mathcal{L}_D$ *contains* $x$ *and the size of* $\mathcal{L}_D$ *is at most* $2^{a+d+r} \cdot L$*.*

*Proof.* $\mathcal{L}_D$ is defined as $\bigcup\{\, R^D(\alpha,z,w) \mid \alpha \in \{0,1\}^a, z \in \{0,1\}^d, w \in \{0,1\}^r \,\}$. □

The entire section is devoted to proving the following result.

**Theorem 4.3.** *For any sufficiently large* $n, m \in \mathbb{N}$ *such that* $m \leq 2n$*, there exists a BB-HSG* $(H, A, R)$ *such that*

$$\begin{aligned} H&\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m,\\ A&\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m,\\ R^{(\text{-})}&\colon \{0,1\}^m \times \{0,1\}^d \times \{0,1\}^r \to (\{0,1\}^n)^L, \end{aligned}$$

*where the advice complexity is* $a := m$*, seed length is* $d = O(\log n + \log^3 m)$*, randomness complexity is* $r = m + O(\log n)$*, advantage is* $\epsilon = 1/m$*, error probability is* $\delta = o(1)$*, and list complexity is* $L = n^{O(1)}$*.*

The significance of Theorem 4.3 is that $2^{a+d+r} \cdot L \leq \mathsf{poly}(n)$ holds for $m = O(\log n)$. By Proposition 4.2, this means that, using an oracle $D$, one can efficiently compute a list of $\mathsf{poly}(n)$ strings that contains all strings $x \in \{0,1\}^n$ such that $D$ $\epsilon$-avoids $H(x,\text{-})$. More generally, we obtain the following corollary.

**Corollary 4.4.** *For any sufficiently large $n, m \in \mathbb{N}$ such that $m \leq 2n$, there exists a function $H\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$, where $d = O(\log n + \log^3 m)$, such that, for every function $D\colon \{0,1\}^m \to \{0,1\}$, there exists a list $\mathcal{L}_D \subseteq \{0,1\}^n$ of $n$-bit strings such that $x \in \mathcal{L}_D$ holds for every $x \in \{0,1\}^n$ such that $D$ avoids $H(x,\text{-})$, and $|\mathcal{L}_D| \leq 2^{2m+O(\log n+\log^3 m)}$. Moreover, the function $H$ can be computed in time $\mathsf{poly}(n)$, and each element of the list $\mathcal{L}_D$ can be computed in time $\mathsf{poly}(n)$ given the index of the element as input and oracle access to $D$.*

## 4.1 Composition Theorem

Following [Uma09], it is useful to consider an advice function $A\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^a$ as a compressor that takes $n$-bit input and compresses it to an $a$-bit string, ignoring the $d$-bit seed (i.e., $d := 0$); similarly, a reconstruction function $R^{(\text{-})}\colon \{0,1\}^a \times \{0,1\}^d \times \{0,1\}^r \to (\{0,1\}^n)^L$ can be regarded as a decompressor when $d = r = 0$, $L = 1$. In this simplified case, we define the composition of BB-HSGs $(H_1, A_1\colon \{0,1\}^n \to \{0,1\}^{a_1}, R_1\colon \{0,1\}^{a_1} \to \{0,1\}^n)$ and $(H_2, A_2\colon \{0,1\}^{a_1} \to \{0,1\}^{a_2}, R_2\colon \{0,1\}^{a_2} \to \{0,1\}^{a_1})$ as the BB-HSG $(H_2, A_2, R_2) \circ (H_1, A_1, R_1) := (H, A_2 \circ A_1, R_1 \circ R_2)$ for some $H$. The general definition is given below.

**Theorem 4.5** (Composition Theorem)**.** *Assume that, for each $i \in \{1,2\}$, $(H_i, A_i, R_i)$ is a BB-HSG such that*

$$\begin{aligned} H_i &\colon \{0,1\}^{n_i} \times \{0,1\}^{d_i} \to \{0,1\}^m, \\ A_i &\colon \{0,1\}^{n_i} \times \{0,1\}^{d_i} \to \{0,1\}^{a_i}, \\ R_i^{(\text{-})} &\colon \{0,1\}^{a_i} \times \{0,1\}^{d_i} \times \{0,1\}^r \to (\{0,1\}^{n_i})^{L_i}, \end{aligned}$$

*with advantage $\epsilon$ and error probability $\delta_i$, where $a_1 = n_2$. Define the* composition *of $(H_1, A_1, R_1)$ and $(H_2, A_2, R_2)$ to be $(H, A, R)$, denoted by $(H_2, A_2, R_2) \circ (H_1, A_1, R_1)$, where $d := d_1 + d_2 + 1$ and*

$$\begin{aligned} H &\colon \{0,1\}^{n_1} \times \{0,1\}^{d} \to \{0,1\}^m, \\ A &\colon \{0,1\}^{n_1} \times \{0,1\}^{d} \to \{0,1\}^{a_2}, \\ R^{(\text{-})} &\colon \{0,1\}^{a_2} \times \{0,1\}^{d} \times \{0,1\}^r \to (\{0,1\}^{n_1})^{L_1 L_2}, \end{aligned}$$

*are defined as*

$$\begin{aligned} H(x, (z_1, z_2, 0)) &:= H_1(x, z_1), \\ H(x, (z_1, z_2, 1)) &:= H_2(A_1(x, z_1), z_2) \\ A(x, (z_1, z_2, i)) &:= A_2(A_1(x, z_1), z_2), \\ R^D(\alpha, (z_1, z_2, i), w) &:= \{\, x' \in R_1^D(y', z_1, w) \mid y' \in R_2^D(\alpha, z_2, w) \,\} \end{aligned}$$

*for any $z_1 \in \{0,1\}^{d_1}, z_2 \in \{0,1\}^{d_2}$ and $i \in \{0,1\}$. Then, $(H, A, R)$ is a BB-HSG with advantage $\epsilon$ and error probability $\delta_1 + 2^{d_1}\delta_2$.*

We emphasize that the randomness $w \in \{0,1\}^r$ of $R$ can be reused in the composition above.

*Proof.* Fix an arbitrary string $x \in \{0,1\}^n$. Assume that a function $D\colon \{0,1\}^m \to \{0,1\}$ $\epsilon$-avoids $H(x,\text{-})\colon \{0,1\}^d \to \{0,1\}^m$. Since $\mathrm{Im}(H(x,\text{-})) = \mathrm{Im}(H_1(x,\text{-})) \cup \mathrm{Im}(H_2(A_1(x,\text{-}),\text{-}))$, $D$ $\epsilon$-avoids $H_1(x,\text{-})$ and $H_2(A_1(x,z_1),\text{-})$ for every $z_1 \in \{0,1\}^{d_1}$.

Therefore,

$$\Pr_w\left[\forall z_1,\ x \notin R_1^D(A_1(x,z_1),z_1,w)\right] \le \delta_1, \text{ and}$$
$$\Pr_w\left[\forall z_2,\ y \notin R_2^D(A_2(y,z_2),z_2,w)\right] \le \delta_2$$

for any $y \in \mathrm{Im}(A_1(x,\text{-}))$.

In order to apply a union bound, we claim the following for any $w \in \{0,1\}^r$.

**Claim 4.6.** *Assume that $x \notin R^D(A(x,z),z,w)$ for any $z = (z_1,z_2,i) \in \{0,1\}^{d_1+d_2+1}$. Then, either $\forall z_1, x \notin R_1^D(A_1(x,z_1),z_1,w)$ or $\exists z_1, \forall z_2, y_{z_1} \notin R_2^D(A_2(y_{z_1},z_2),z_2,w)$, where $y_{z_1} := A_1(x,z_1)$.*

We prove the contrapositive of Claim 4.6. Assume that $\exists z_1, x \in R_1^D(A_1(x,z_1),z_1,w)$ and $\forall z_1', \exists z_2, y_{z_1'} \in R_2^D(A_2(y_{z_1'},z_2),z_2,w)$. Take the $z_1$ in the first statement and define $z_1' := z_1$ and $y := y_{z_1} = A_1(x,z_1)$. Since

$$A(x,z) = A_2(A_1(x,z_1),z_2) = A_2(y,z_2),$$

we have

$$R^D(A(x,z),z,w) = \{\, x' \in R_1^D(y',z_1,w) \mid y' \in R_2^D(A_2(y,z_2),z_2,w) \,\},$$

which contains $x$ because the condition is satisfied for $x' := x$ and $y' := y$. This completes the proof of Claim 4.6.

By using a union bound, we obtain

$$\begin{aligned}
&\Pr_w\left[\forall z,\ x \notin R^D(A(x,z),z,w)\right] \\
&\le \Pr_w\left[\forall z_1,\ x \notin R_1^D(A_1(x,z_1),z_1,w)\right] + \Pr_w\left[\exists z_1, \forall z_2,\ y_{z_1} \notin R_2^D(A_2(y_{z_1},z_2),z_2,w)\right] \\
&\le \delta_1 + \sum_{y \in \mathrm{Im}(A_1(x,\text{-}))} \Pr_w\left[\forall z_2,\ y \notin R_2^D(A_2(y,z_2),z_2,w)\right] \\
&\le \delta_1 + 2^{d_1}\delta_2,
\end{aligned}$$

as desired. □

## 4.2 Basic Constructions of BB-HSGs

We now describe two constructions of BB-HSGs that are building blocks for our final construction.

**Theorem 4.7.** *For any constant $\alpha > 0$ and any sufficiently large parameters $n, m, \delta^{-1} \in \mathbb{N}$ such that $m \le 2n$, there exists a BB-HSG $(H, A, R)$ such that*

$$H\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m,$$
$$A\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^a,$$
$$R^{(\text{-})}\colon \{0,1\}^a \times \{0,1\}^d \times \{0,1\}^r \to (\{0,1\}^n)^L,$$

*with randomness complexity $r = m + O(\log(1/\delta))$, advantage $\epsilon = 1/m$, error probability $\delta$, list complexity $L = \mathsf{poly}(m) \cdot \log(1/\delta)$, and one of the following parameter settings:*

1. $d = O(\log n)$ *and* $a \leq n^{\alpha} m$.

2. $d = O(\log^3 n)$ *and* $a \leq m$.

Both constructions are based on an improvement of Trevisan's extractor [Tre01] given by Raz, Reingold, and Vadhan [RRV02]. We will modify the constructions using the randomness efficient construction of a hitter so that the error probability is reduced to $\delta$. The following lemma reduces the error probability.

**Lemma 4.8.** *Let $(H, A, R_0)$ be a BB-HSG with error probability $1-\epsilon_0$, randomness complexity $r_0$, and list complexity $L_0$. Then, there exists a BB-HSG $(H, A, R)$ with error probability $\delta$, randomness complexity $r := r_0 + O(\log(1/\delta))$, and list complexity $L = O(L_0 \cdot \log(1/\delta)/\epsilon_0)$.*

This lemma is an immediate consequence of the existence of a hitter.

**Lemma 4.9** (Hitter; see [Gol11, Appendix C])**.** *For any parameters $n \in \mathbb{N}, \epsilon, \delta > 0$, there exists an efficiently computable function* Hit$\colon \{0,1\}^{n+O(\log(1/\delta))} \times [L] \to \{0,1\}^n$ *(called a* hitter*) such that $L = O(\log(1/\delta)/\epsilon)$ and for any subset $H \subseteq \{0,1\}^n$ of size at least $\epsilon 2^n$,*

$$\Pr_z \left[\exists k \in [L],\ \mathrm{Hit}(z,k) \in H\right] \geq 1 - \delta.$$

*Proof of Lemma 4.8.* Using the hitter of Lemma 4.9, we define $R^D(\alpha, z, w) := \bigcup_{k \in [L']} R_0^D(\alpha, z, \mathrm{Hit}(w,k))$ for any $\alpha \in \{0,1\}^a, z \in \{0,1\}^d, w \in \{0,1\}^r$, where $L' = O(\log(1/\delta)/\epsilon_0)$ and $r = r_0 + O(\log(1/\delta))$. By the property of the BB-HSG $(H, A, R_0)$, for any $x \in \{0,1\}^n$ and any oracle $D\colon \{0,1\}^m \to \{0,1\}$ that $\epsilon$-avoids $H(x, \text{-})$, we have

$$\Pr_{w_0 \sim \{0,1\}^{r_0}} \left[\exists z \in \{0,1\}^d,\ x \in R_0^D(A(x,z), z, w_0)\right] \geq \epsilon_0.$$

It follows from the property of Hit that

$$\Pr_{w \sim \{0,1\}^{r}} \left[\exists k \in [L'],\ \exists z \in \{0,1\}^d,\ x \in R_0^D(A(x,z), z, \mathrm{Hit}(w,k))\right] \geq 1 - \delta.$$

Equivalently,

$$\Pr_{w \sim \{0,1\}^{r}} \left[\exists z \in \{0,1\}^d,\ x \in R^D(A(x,z), z, w)\right] \geq 1 - \delta.$$

□

We now describe the construction of Trevisan's extractor [Tre01, RRV02]. Let us first recall the notion of weak design.

**Lemma 4.10** (Weak design [RRV02])**.** *For any constant $\alpha > 0$ and any sufficiently large parameters $d, \ell, m \in \mathbb{N}$, there exists a "*weak design*" $S_1, \cdots, S_m \subseteq [d]$ such that $|S_i| = \ell$ and $\sum_{j<i} 2^{|S_i \cap S_j|} \leq \rho \cdot (m-1)$ for any $i \in [m]$ if either*

1. *$\rho := 2^{\alpha \ell}$ and $d = O(\ell/\alpha)$, or*

2. *$\rho := 1$ and $d = O(\ell^2 \cdot \log m)$.*

*Moreover, the weak design can be computed in time* $\mathsf{poly}(m, d)$ *given the parameters* $(d, \ell, m, \rho)$ *as input.*

We also require a list-decodable error-correcting code, which can be constructed by concatenating the Reed-Solomon code and Hadamard code.

**Lemma 4.11** ([Sud97]; see, e.g., [Vad12])**.** *For any parameters* $n, \epsilon$*, there exists a function* $\mathrm{Enc}\colon \{0,1\}^n \to \{0,1\}^{\widehat{n}}$ *such that*

- $\widehat{n} = 2^\ell$ *for some integer* $\ell \in \mathbb{N}$ *and* $\widehat{n} \le \mathsf{poly}(n, 1/\epsilon)$*,*
- $\mathrm{Enc}$ *is computable in time* $\mathsf{poly}(n, 1/\epsilon)$*, and*
- *given* $y \in \{0,1\}^{\widehat{n}}$*, one can find a list of size* $\mathsf{poly}(1/\epsilon)$ *that contains all the strings* $x \in \{0,1\}^n$ *such that* $y$ *and* $\mathrm{Enc}(x)$ *agree on at least a* $(1/2+\epsilon)$*-fraction of coordinates.*

For a string $z \in \{0,1\}^d$ and a subset $S \subseteq [d]$, let $z_S$ denote $(z_i)_{i \in S} \in \{0,1\}^S$, i.e., the $|S|$-bit string that can be obtained by concatenating $z_i$ for all $i \in S$ (in increasing order of $i$).

We are now ready to describe the construction of Theorem 4.7.

**HSG function** $H\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$**.**

For $x \in \{0,1\}^n$ and $z \in \{0,1\}^d$, we define $H(x,z)$ as follows. Take the error-correcting code $\mathrm{Enc}\colon \{0,1\}^n \to \{0,1\}^{2^\ell}$ of Lemma 4.11 with the parameter $\epsilon_0 := 1/2m^2$; then, we have $\ell = O(\log n + \log m) = O(\log n)$. Let $\widehat{x}\colon \{0,1\}^\ell \to \{0,1\}$ be the function whose truth table is $\mathrm{Enc}(x) \in \{0,1\}^{2^\ell}$ (i.e., $\widehat{x}$ maps $z \in \{0,1\}^\ell$ to the $z$th bit of $\mathrm{Enc}(x)$). Take the weak design $S_1, \cdots, S_m \subseteq [d]$ of Lemma 4.10. Then, $H(x,z)$ is defined as $(\widehat{x}(z_{S_i}) \mid i \in [m])$.

**Advice function** $A\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^a$**.**

For $x \in \{0,1\}^n$ and $z' \in \{0,1\}^d$, the advice $A(x,z')$ is defined as follows. Regard the seed $z' \in \{0,1\}^d$ as $(i, z_{[d]\setminus S_i}) \in [m] \times \{0,1\}^{d-\ell}$. (This is possible because $\lceil \log m \rceil + d - \ell \le d$.) Each bit of the advice $A(x,z')$ is indexed by the set $\mathcal{I}_i := \{(j, z_{S_i \cap S_j}) \in [i-1] \times \{0,1\}^{|S_i \cap S_j|}\}$. The size of $\mathcal{I}_i$ is at most $\sum_{j<i} 2^{|S_i \cap S_j|} \le \rho \cdot (m-1) =: a$. For each $(j, z_{S_i \cap S_j}) \in \mathcal{I}_i$, we write down $\widehat{x}(z_{S_j})$ as one bit of $A(x,z')$. That is, $A(x,z') := \big(\widehat{x}(z_{S_j}) \mid (j, z_{S_i \cap S_j}) \in \mathcal{I}_i\big)$.

**Reconstruction function** $R_0^D\colon \{0,1\}^a \times \{0,1\}^d \times \{0,1\}^m \to (\{0,1\}^n)^{L_0}$**.**

We describe a basic reconstruction procedure $R_0^D$ with error probability $1 - \epsilon_0$, where $\epsilon_0 := 1/2m^2$. By using Lemma 4.8, we convert $R_0$ to $R\colon \{0,1\}^a \times \{0,1\}^d \times \{0,1\}^{m+O(\log(1/\delta))} \to (\{0,1\}^n)^L$, whose list complexity is $L = O((L_0 \log(1/\delta))/\epsilon_0)$.

$R_0^D$ is defined as follows. Given $\alpha \in \{0,1\}^a, (i, z_{[d]\setminus S_i}) \in [m] \times \{0,1\}^{d-\ell}$, and $w \in \{0,1\}^m$ as input, we identify the advice $\alpha$ with the function $\alpha\colon \mathcal{I}_i \to \{0,1\}$ by identifying $\mathcal{I}_i$ with $[a]$. We define $y\colon \{0,1\}^\ell \to \{0,1\}$ as

$$y(z_{S_i}) := D(\alpha(1, z_{S_i \cap S_1}), \ldots, \alpha(i-1, z_{S_i \cap S_{i-1}}), w_i, w_{i+1}, \ldots, w_m) \oplus w_i$$

for any $z_{S_i} \in \{0,1\}^\ell$. The output of $R_0^D$ is defined as the list of all strings $x \in \{0,1\}^n$ such that $\widehat{x}$ and $y$ agree on a $(1/2+\epsilon_0)$-fraction of inputs. This can be efficiently computed using the list-decoding algorithm of Lemma 4.11. The list complexity $L_0$ is $\mathsf{poly}(1/\epsilon_0) = m^{O(1)}$. The list complexity $L$ of the final reconstruction procedure $R$ is $O((L_0 \log(1/\delta))/\epsilon_0) = m^{O(1)} \cdot \log(1/\delta)$.

The correctness of the construction $(H, A, R_0)$ is ensured by the following lemma.

**Lemma 4.12** (Yao [Yao82]; Nisan and Wigderson [NW94])**.** *The construction $(H, A, R_0)$ described above is a BB-HSG with advantage $\epsilon = 1/m$ and error probability $1 - \epsilon_0$.*

*Proof.* Fix a string $x \in \{0,1\}^n$ and an oracle $D$ that $\epsilon$-avoids $H(x,\text{-})$. We have $\Pr_w[D(w) = 1] \geq \epsilon$ and $D(H(x,z)) = 0$ for every $z \in \{0,1\}^d$. For $i \in \{0,\ldots,m\}$, define the $i$th hybrid distribution as $H_i \equiv \widehat{x}(z_{S_1}) \cdots \widehat{x}(z_{S_i}) \cdot w_{i+1} \cdots w_m$ with $z \sim \{0,1\}^d$ and $w \sim \{0,1\}^m$. Since $H_0 \equiv w$ and $H_m \equiv H(x,z)$, we obtain $\mathbb{E}[D(H_0) - D(H_m)] \geq \epsilon$. Choosing $i \sim [m]$ randomly,

$$\begin{aligned}\frac{\epsilon}{m} &\leq \mathbb{E}[D(H_{i-1}) - D(H_i)] \\ &= \frac{1}{2}\mathbb{E}[D(H_i)] + \frac{1}{2}\mathbb{E}[D(H_{i-1}) \mid w_i \neq \widehat{x}(z_{S_i})] - \mathbb{E}[D(H_i)] \\ &= \frac{1}{2}\mathbb{E}[D(H_{i-1}) \mid w_i \neq \widehat{x}(z_{S_i})] - \frac{1}{2}\mathbb{E}[D(H_i)],\end{aligned}$$

where the first equality uses the case analysis on whether $w_i = \widehat{x}(z_{S_i})$ or not. Similarly,

$$\begin{aligned}\Pr[D(H_{i-1}) \oplus w_i = \widehat{x}(z_{S_i})] &= \frac{1}{2}\Pr[D(H_i) = 0] + \frac{1}{2}\Pr[D(H_{i-1}) = 1 \mid w_i \neq \widehat{x}(z_{S_i})] \\ &= \frac{1}{2} + \frac{1}{2}\mathbb{E}[D(H_{i-1}) \mid w_i \neq \widehat{x}(z_{S_i})] - \frac{1}{2}\mathbb{E}[D(H_i)] \\ &\geq \frac{1}{2} + \frac{\epsilon}{m} = \frac{1}{2} + 2\epsilon_0.\end{aligned}$$

Therefore,

$$\begin{aligned}\Pr_w\left[\exists z', x \in R_0^D\big(A(x,z'), z', w\big)\right] &\geq \Pr_{w,z'}\left[x \in R_0^D\big(A(x,z'), z', w\big)\right] \\ &\geq \Pr_{w,i,z_{[d]\setminus S_i}}\left[\Pr_{z_{S_i}}[D(H_{i-1}) \oplus w_i = \widehat{x}(z_{S_i})] \geq \frac{1}{2} + \epsilon_0\right] \quad \geq \quad \epsilon_0.\end{aligned}$$

□

Choosing the parameter $\rho$ appropriately and combining Lemma 4.12 and Lemma 4.8, we obtain Theorem 4.7. Specifically, we have the following.

1. If we choose $\rho := 2^{\alpha\ell}$ as the parameter of the weak design in Lemma 4.10, the advice complexity is $a = \rho \cdot (m-1) = 2^{\alpha\ell} \cdot (m-1) \leq n^{O(\alpha)}m$, where $O(\text{-})$ hides some universal constant that does not depend on $\alpha > 0$. By choosing $\alpha > 0$ as a small constant, the seed length is $d = O(\ell/\alpha) = O(\log n)$.

2. If we choose $\rho := 1$, the advice complexity is $a = \rho \cdot (m-1) = m - 1$ and the seed length is $d = O(\ell^2 \cdot \log m) = O(\log^3 n)$.

## 4.3 Composing Basic Constructions

We now compose the basic construction given in Item 1 of Theorem 4.7 with itself several times, and reduce the advice complexity $a$ to $m^3$. Then we will compose it with Item 2 of Theorem 4.7

and further reduce the advice complexity from $m^3$ to $m$, which will complete the construction of Theorem 4.3.

The following lemma composes Item 1 of Theorem 4.7 with itself approximately $O(\log \log n)$ times.

**Lemma 4.13.** *For any sufficiently large parameters $n, m \in \mathbb{N}$ such that $m^2 \leq n$, there exists a BB-HSG $(H\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m, A, R)$ with advice complexity $a = m^3$, randomness complexity $r = m + O(\log n)$, seed length $d = O(\log n)$, advantage $\epsilon = 1/m$, error probability $\delta = o(1)$, and list complexity $L = \mathsf{poly}(n)$.*

*Proof.* Fix a parameter $m$. Let $\alpha := 1/2$. We define a sequence $\{n_k\}_{k\in\mathbb{N}}$ as $n_0 := n$ and $n_{k+1} = n_k^\alpha \cdot m$ for any $k \in \mathbb{N}$. Note that

$$n_k = n_{k-1}^\alpha \cdot m = n_{k-2}^{\alpha^2} \cdot m^\alpha \cdot m = \cdots \leq n_0^{\alpha^k} \cdot m^{1+\alpha+\alpha^2+\cdots} = n^{\alpha^k} \cdot m^{1/(1-\alpha)}.$$

For each $k \in \mathbb{N}$, let $(H_k, A_k\colon \{0,1\}^{n_k} \times \{0,1\}^{d_k} \to \{0,1\}^{a_k}, R_k)$ be the BB-HSG construction of Item 1 of Theorem 4.7, where $a_k = n_k^\alpha \cdot m = n_{k+1}$, $d_k = c \log n_k$, and list complexity $L_k = m^c \cdot \log(1/\delta')$ for some universal constant $c > 0$. Choose the error parameter $\delta' := 1/n^{c+1}$. Applying the Composition Theorem, we inductively define $(H_{\leq 0}, A_{\leq 0}, R_{\leq 0}) := (H_0, A_0, R_0)$ and $(H_{\leq k+1}, A_{\leq k+1}, R_{\leq k+1}) := (H_{k+1}, A_{k+1}, R_{k+1}) \circ (H_{\leq k}, A_{\leq k}, R_{\leq k})$ for any $k \in \mathbb{N}$. This is well defined because the advice complexity $a_{\leq k}$ of $A_{\leq k}$ is equal to $a_k$; thus, $A_{\leq k}$ can be composed with $A_{k+1}$, whose input length $n_{k+1}$ is equal to $a_k$.

We stop the process at $K := \log(\log n / \log m) \leq \log n / \log m$ and define $(H, A, R) := (H_{\leq K}, A_{\leq K}, R_{\leq K})$. The advice complexity of $(H, A, R)$ is $a := a_K = n_{K+1} \leq n^{2^{-K-1}} \cdot m^2 \leq m^3$, seed length is $d = d_{\leq K} = d_K + d_{\leq K-1} + 1 = \sum_{k=0}^K d_k + K \leq \sum_{k=0}^K (c\alpha^k \log n + O(\log m)) = O(\log n)$, error probability is $\delta = \delta_{\leq K} = \delta_{\leq K-1} + 2^{d_{\leq K-1}}\delta' = \delta' \sum_{k=1}^K 2^{d_{\leq k-1}} \leq n^{-c-1} K n^c = o(1)$, and list complexity is $L = L_{\leq K} = L_K L_{\leq K-1} = \prod_{k=0}^K L_k = m^{cK} (\log(1/\delta'))^K \leq n^{O(1)}$. □

Finally, we compose the BB-HSG of Lemma 4.13 with Item 2 of Theorem 4.7.

*Proof of Theorem 4.3.* If $m^2 \geq n$, the BB-HSG of Item 2 of Theorem 4.7 achieves the parameters stated in Theorem 4.3. Otherwise, let $(H_1\colon \{0,1\}^n \times \{0,1\}^{d_1} \to \{0,1\}^m, A_1, R_1)$ be the BB-HSG of Lemma 4.13. Let $(H_2\colon \{0,1\}^{m^3} \times \{0,1\}^{d_2} \to \{0,1\}^m, A_2, R_2)$ be the BB-HSG of Item 2 of Theorem 4.7 for $\delta_2 = o(2^{-d_1})$. Note that the advice complexity of $(H_1, A_1, R_1)$ is $m^3$ and that of $(H_2, A_2, R_2)$ is $m$. Thus, using the Composition Theorem, we can define $(H, A, R) := (H_2, A_2, R_2) \circ (H_1, A_1, R_1)$, whose advice complexity is $m$, seed length is $d_1 + d_2 + 1 = O(\log n + \log^3 m)$, error probability is $\delta_1 + 2^{d_1}\delta_2 = o(1)$, and list complexity is $n^{O(1)}$. □

# 5 Hardness Under Deterministic Reductions

Using the BB-HSG of Theorem 4.3, we present the $\mathsf{E}^{\mathsf{NP}}$-hardness result of GapMINKT$^{\mathsf{NP}}$. It is possible to provide a proof of the $\mathsf{E}^{\mathsf{NP}}$-hardness as a corollary of the non-black-box worst-case-to-average-case reduction given in Theorem 8.1, as explained in Section 2.1.3; however, here we present a self-contained and succinct proof.

**Theorem 5.1.** *For any $\mathsf{NP}$-hard oracle $A$ and any polynomial $\tau\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$, $\mathsf{E}^{\mathsf{NP}}$ is reducible to* Gap$_\tau$MINKT$^A$ *via a deterministic nonadaptive $\mathsf{E}$-reduction.*

*Proof.* Let $L \in \mathsf{E}^{\mathsf{NP}}$. Then, there exist a language $S \in \mathsf{NP}$ with an $n^c$-time $\mathsf{NP}$-verifier $V$ and a $t(n)$-time oracle machine $M$ such that $M^S(x) = L(x)$ for every $x \in \{0,1\}^*$ and $t(n) = 2^{O(n)}$, where $c$ is some constant.

Fix any input $x \in \{0,1\}^*$ of length $n$. Define a function $f_x \colon [t(n)] \to \{0,1\}^{t(n)^c}$ such that, for each $i \in [t(n)]$, $f_x(i)$ is the lexicographically first $\mathsf{NP}$-certificate for $q \in S$ (we define $f_x(i) := 1^{t(n)^c}$ if $q \notin S$), where $q$ is the $i$th query that $M^S$ makes on input $x$. Note that the truth table of $f_x$ can be computed in time $2^{O(n)}$ given an input $x$ and access to the $\mathsf{NP}$-hard oracle $A$.

Let $B$ be an oracle that is consistent with $\mathrm{Gap}_\tau \mathrm{MINKT}^A$. The proof consists of two parts.

**Claim 5.2.** *1. Given nonadaptive oracle access to $B$, one can efficiently compute the list of $2^{O(n)}$ functions $f' \colon [t(n)] \to \{0,1\}^{t(n)^c}$, one of which is guaranteed to be $f_x$.*

*2. Given a list $\mathcal{L}$ of $2^{O(n)}$ functions $f'$, one of which is guaranteed to be $f_x$, one can compute $L$ in time $2^{O(n)}$.*

It is clear that Theorem 5.1 follows from Claim 5.2; we briefly note that the idea behind Claim 5.2 is that $\mathsf{E}^{\mathsf{NP}}$ has an exponential-time selector [BH92, Hir15].

We first prove Item 2. Let $\mathcal{L}$ be a list of functions such that $f_x \in \mathcal{L}$. We can simulate the query $q$ of the machine $M^S$ as follows. Let $i \in [t(n)]$ be the index of the query $q$. Observe that $q \in S$ if and only if there exists a function $f \in \mathcal{L}$ such that $f(i)$ is a certificate for $q \in S$. Indeed, $f_x$ is guaranteed to be in the list $\mathcal{L}$, and $f_x(i)$ is a certificate for $q \in S$ if any. Therefore, we answer YES to the $i$th query $q$ if and only if $V(q, f(i))$ accepts for some $f \in \mathcal{L}$. This simulation takes time $2^{O(n)}$, which completes the proof of Item 2.

It remains to prove Item 1. Identify $f_x \colon [t(n)] \to \{0,1\}^{t(n)^c}$ with a string $f_x \in \{0,1\}^{t(n)^{c+1}}$ in a canonical manner. Let $N := t(n)^{c+1} = 2^{O(n)}$, $m = O(n)$ be a parameter chosen later, and $H \colon \{0,1\}^N \times \{0,1\}^d \to \{0,1\}^m$ be the BB-HSG from Corollary 4.4. Since $d = O(\log N + \log^3 m)$, there exists some universal constant (independent of the choice of $m$) such that $d \le c_1 n$ for all large $n$. We claim that $\mathrm{K}^{t'(n),A}(H(f_x, z)) \le |z| + |x| + O(\log n)$ for every $z \in \{0,1\}^d$, where $t'(n) = 2^{c_2 n}$ for some constant $c_2$ independent of $m$. Indeed, the output of $H(f_x, z)$ can be described with $n$, $x \in \{0,1\}^n$, and $z \in \{0,1\}^d$ by computing $f_x$ with the $\mathsf{NP}$-hard oracle $A$ in time $2^{O(n)}$ and then computing $H(f_x, z)$ in time $\mathsf{poly}(N) = 2^{O(n)}$; therefore, for all large $n$,

$$\mathrm{K}^{t'(n),A}(H(f_x, z)) \le d + n + O(\log n) \le (c_1 + 2)n.$$

We now choose $m = O(n)$ large enough so that $(c_1+2)n + \log \tau(m, t'(n)) < m/2$. A simple counting argument shows that $\mathrm{K}^{\tau(m,t'(n)),A}(w) \ge \mathrm{K}^A(w) \ge m/2$ for a $(1 - o(1))$-fraction of $w \in \{0,1\}^m$. In particular, the oracle $B$ rejects $(w, 1^{t'(n)}, 1^{(c_1+2)n})$ for most $w \in \{0,1\}^m$, whereas $B$ accepts $(H(f_x, z), 1^{t'(n)}, 1^{(c_1+2)n})$ for every $z \in \{0,1\}^d$. This means that $H(f_x, \text{-})$ is $\frac{1}{2}$-avoided by the function $D \colon \{0,1\}^m \to \{0,1\}$ defined as $D(w) := \neg B(w, 1^{t'(n)}, 1^{(c_1+2)n})$. By the reconstruction property of the BB-HSG $H$ (Corollary 4.4), there exists a $D$-oracle $\mathsf{poly}(N)$-time algorithm $E^D$ that outputs a list $\mathcal{L}_D$ such that $f_x \in \mathcal{L}_D$. The $D$-oracle algorithm $E^D$ can be modified to a $B$-oracle algorithm $E'^B$ by defining $E'^B(x) := E^{B(\text{-}, 1^{t'(n)}, 1^{(c_1+2)n})}(x)$. Finally, observe that $E'^B$ and $E^D$ are nonadaptive oracle algorithms since there are $2^m$ possible queries to $D \colon \{0,1\}^m \to \{0,1\}$, all of which can be asked beforehand in time $2^{O(m)} = 2^{O(n)}$. □

We emphasize that the reduction of Theorem 5.1 is nonadaptive because the number of possible queries is small. In contrast, the reconstruction procedure of the BB-HSG of Theorem 4.3 itself is adaptive because of the composition of the BB-HSGs.

Since the number of possible queries is small, we can easily extend Theorem 5.1 to the case of average-case assumptions.

**Theorem 5.3.** *Let $A$ be any* NP*-oracle. If* $\{\mathsf{coMINKT}^A\} \times \mathsf{PSamp} \subseteq \mathsf{Avg}^1_{1-2^{-n}}\mathsf{P}$*, then* $\mathsf{E}^{\mathsf{NP}} = \mathsf{E}$.

*Proof.* Let $L \in \mathsf{E}^{\mathsf{NP}}$. For each $n \in \mathbb{N}$, let $Q_n$ denote the set of queries that the E-reduction (to $\mathrm{GapMINKT}^A$) of Theorem 5.1 makes. By inspecting the proof of Theorem 5.1, one can observe that $Q_n \subseteq \{\, (x, 1^{t(|x|)}, 1^{s(|x|)}) \mid x \in \{0,1\}^{O(n)} \,\}$ for some functions $t(n') = 2^{O(n')}$ and $s(n') = O(n')$.

We define the family of distributions $\mathcal{D} = \{\mathcal{D}_n\}_{n\in\mathbb{N}}$ as follows. For each $n \in \mathbb{N}$, let $x_n := (\mathsf{bin}(n), 1^{t(|\mathsf{bin}(n)|)}, 1^{s(|\mathsf{bin}(n)|)})$, where $\mathsf{bin}(n)$ denotes the $n$th string in $\{0,1\}^*$, and define the distribution $\mathcal{D}_n$ such that $\mathsf{supp}(\mathcal{D}_n) = \{x_n\}$. The length of $x_n$ is $|x_n| = O(\log n) + 2^{O(\log n)} \leq n^{O(1)}$; thus, the distribution $\mathcal{D}$ is polynomial-time samplable.

Let $L := \mathsf{coMINKT}^A$. By the assumption, there exists a polynomial-time one-sided-error heuristic algorithm $M$ for $(\mathsf{coMINKT}^A, \mathcal{D})$ with success probability $2^{-n}$ on inputs from $\mathcal{D}_n$ for any $n \in \mathbb{N}$. By the definition of one-sided-error heuristic algorithms, we have $\Pr_{x\sim\mathcal{D}_n}[M(x) = L(x)] \geq 2^{-n} > 0$ for every $n \in \mathbb{N}$. Since $\mathcal{D}_n$ is supported only on $\{x_n\}$, we obtain $M(x_n) = L(x_n)$. This means that $M$ solves $L = \mathsf{coMINKT}^A$ on any input in $\{\, x_n \mid n \in \mathbb{N} \,\}$. Finally, observe that $\bigcup_{n\in\mathbb{N}} Q_n \subseteq \{\, x_n \mid n \in \mathbb{N} \,\}$, from which it follows that $\neg M$ solves $\mathrm{MINKT}^A$ on $\bigcup_{n\in\mathbb{N}} Q_n$; by running the reduction of Theorem 5.1 with the oracle $\neg M$, we obtain $\mathsf{E}^{\mathsf{NP}} = \mathsf{E}$. □

# 6 Constructing Pseudorandom Generators

In this section, we construct a nearly optimal pseudorandom generator under the assumption that any one of Items 1 to 9 of Theorem 1.14 is true. (We defer the corresponding results for Items 10 to 12 to Section 8.)

## 6.1 Exposition of the PRG Construction from Heuristics for NP

We review the pseudorandom generator construction of Buhrman, Fortnow, and Pavan [BFP05] from $\mathsf{DistNP} \subseteq \mathsf{AvgP}$, and then we observe that the construction works even under the assumption that $\mathsf{Dist}(\mathsf{coNP})$ admits a one-sided-error heuristic algorithm with low success probability.

**Theorem 6.1** ([BFP05])**.** *If* $\mathsf{DistNP} \subseteq \mathsf{AvgP}$*, then there exists a nearly optimal pseudorandom generator.*

The proof of Theorem 6.1 relies on the following two results.

1. If $\mathsf{DistNP} \subseteq \mathsf{AvgP}$, then $\mathsf{E} = \mathsf{NE}$ [BCGL92].
2. If $\mathsf{DistNP} \subseteq \mathsf{AvgP}$, then $\text{pr-}\mathsf{MA} = \mathsf{NP}$ [KS04].

These results are sufficient to obtain a nearly optimal pseudorandom generator, as shown in the following lemma.

**Lemma 6.2** (Implicit in [BFP05])**.** *Assume that* $\mathsf{E} = \mathsf{NE}$ *and* $\text{pr-}\mathsf{MA} = \mathsf{NP}$*. Then, there exists a nearly optimal pseudorandom generator* $G := \{G_n : \{0,1\}^{O(\log n)} \to \{0,1\}^n\}_{n\in\mathbb{N}}$ *(secure against linear-sized circuits).*

*Proof Sketch.* By the theorem of Impagliazzo and Wigderson [IW97], it suffices to show that $\mathsf{E} \not\subseteq \mathsf{i.o.SIZE}(2^{\epsilon n})$ for some constant $\epsilon > 0$. Here, for a complexity class $\mathfrak{C}$, $\mathsf{i.o.}\mathfrak{C}$ is defined as the class of languages $L \subseteq \{0,1\}^*$ such that there exists some language $L' \in \mathfrak{C}$ such that $L \cap \{0,1\}^n = L' \cap \{0,1\}^n$ for infinitely many $n \in \mathbb{N}$. Let $\mathsf{i.o.SIZE}(2^{o(n)})$ denote $\bigcap_{\epsilon>0} \mathsf{i.o.SIZE}(2^{\epsilon n})$.

Assume, for a contradiction, that $\mathsf{E} \subseteq \mathsf{i.o.SIZE}(2^{o(n)})$. As in [BFP05], by using an efficient PCP system for $\mathsf{E}$, it follows that $\mathsf{E} \subseteq \mathsf{i.o.MATIME}(2^{o(n)})$. That is, a prover sends a circuit $C$ of size $2^{o(n)}$ that encodes a computation path of $\mathsf{E}$, and then a verifier checks it using a verifier for the PCP system with random access to the circuit $C$.

By applying a padding argument to pr-$\mathsf{MA} = \mathsf{NP}$ (whose proof is given later in Proposition 6.4), we obtain $\mathsf{i.o.MATIME}(2^{o(n)}) = \mathsf{i.o.NTIME}(2^{o(n)})$. Therefore, $\mathsf{E} \subseteq \mathsf{i.o.MATIME}(2^{o(n)}) = \mathsf{i.o.NTIME}(2^{o(n)})$.

However, this contradicts the following Proposition 6.3 and the assumption that $\mathsf{NE} \subseteq \mathsf{E}$. □

**Proposition 6.3** ([IKW02])**.** *If* $\mathsf{NE} \subseteq \mathsf{E}$*, then* $\mathsf{E} \not\subseteq \mathsf{i.o.NTIME}(2^n)$.

*Proof.* We first claim that $\mathsf{NTIME}(2^n) \subseteq \mathsf{DTIME}(2^{2cn})$ for some constant $c$. Take the following canonical $\mathsf{NE}$-complete problem $L$: $(M, x) \in L$ if and only if there exists some nondeterministic path on which the nondeterministic machine $M$ accepts $x$ in time $2^{2|x|}$. Since $L \in \mathsf{NE} \subseteq \mathsf{E}$, there exist a constant $c$ and a $2^{cn}$-time deterministic algorithm that computes $L$ on inputs of length $n$. Now, take any problem $R \in \mathsf{NTIME}(2^n)$, and let $M_R$ be the $O(2^n)$-time nondeterministic machine that decides $R$. Since $R$ is linear-time-reducible to $L$ via the map $x \mapsto (M_R, x)$, we conclude that $R$ can be computed in time $2^{c\cdot(|M_R|+n)} \leq 2^{2cn}$. Therefore, we have $\mathsf{i.o.NTIME}(2^n) \subseteq \mathsf{i.o.DTIME}(2^{2cn})$.

On the other hand, the time hierarchy theorem implies that $\mathsf{E} \not\subseteq \mathsf{i.o.DTIME}(2^{2cn})$; thus, $\mathsf{E} \not\subseteq \mathsf{i.o.NTIME}(2^n)$. □

A similar argument based on a complete problem enables us to use a padding argument to pr-$\mathsf{MA}$, which completes the proof of Lemma 6.2.

**Proposition 6.4.** pr-$\mathsf{MA} = \mathsf{NP}$ *implies* $\mathsf{i.o.MATIME}(2^{o(n)}) = \mathsf{i.o.NTIME}(2^{o(n)})$.

*Proof.* Take a pr-$\mathsf{MA}$-complete problem $\Pi$ under linear-time reductions, which can be constructed by using a Turing machine in a standard manner. Then, $\Pi \in \mathsf{NTIME}(n^k)$ for some fixed constant $k$. Let $\epsilon > 0$ be an arbitrary constant. For any problem $L$ in $\mathsf{MATIME}(2^{\epsilon n})$, let $L' := \{\, x01^{2^{\epsilon n}} \mid x \in L \,\}$ be a padded version of $L$. Then, $L' \in \mathsf{MATIME}(n)$; thus, $L'$ is linear-time reducible to $\Pi \in \mathsf{NTIME}(n^k)$, so $L' \in \mathsf{NTIME}(n^k)$. It follows that $L \in \mathsf{NTIME}(2^{k\epsilon n})$, where $k$ is a constant independent of $\epsilon > 0$. □

## 6.2 PRG from Heuristics with Low Success Probability

It is easy to extend the result of [BFP05] to the case of a one-sided-errorless heuristic algorithm for $\mathsf{Dist}(\mathsf{coNP})$ with low success probability.

**Theorem 6.5.** *If* $\mathsf{Dist}(\mathsf{coNP}) \subseteq \mathsf{Avg}^1_{1-2^{-n}}\mathsf{P}$*, then there exists a nearly optimal pseudorandom generator.*

We observe below that the results of [BCGL92, KS04] can be extended to the case when the assumption is $\mathsf{Dist}(\mathsf{coNP}) \subseteq \mathsf{Avg}^1_{1-2^{-n}}\mathsf{P}$.

**Proposition 6.6** ([BCGL92]). *If* $\mathsf{Dist}(\mathsf{coNP}) \subseteq \mathsf{Avg}^1_{1-2^{-n}}\mathsf{P}$, *then* $\mathsf{E} = \mathsf{NE}$.

*Proof.* Consider the family of distributions $\mathcal{T} := \{\mathcal{T}_n\}_{n\in\mathbb{N}}$ such that $\mathcal{T}_n$ samples a string $1^n$ with probability 1 for each $n \in \mathbb{N}$. It is clear that $\mathcal{T}$ is efficiently samplable.

Let $L \in \mathsf{NE}$. Consider a padded version $L'$ of $L$ defined as $L' := \{\, 1^n \mid n \in \mathbb{N}, \mathsf{bin}(n) \in L \,\}$, where $\mathsf{bin}\colon \mathbb{N} \to \{0,1\}^*$ denotes the bijection that maps an integer $i \in \mathbb{N}$ to the $i$th string of $\{0,1\}^*$ in lexicographical order. It is easy to observe that $\mathsf{co}L' \in \mathsf{coNP}$.

Since $(L', \mathcal{T}) \in \mathsf{Dist}(\mathsf{coNP}) \subseteq \mathsf{Avg}^1_{1-2^{-n}}\mathsf{P}$, there exists some polynomial-time one-sided-error heuristic algorithm $A$ such that $\Pr_{x\sim\mathcal{T}_n}[A(x) = \mathsf{co}L'(x)] \geq 2^{-n}$ for every $n \in \mathbb{N}$. This is equivalent to saying that $A(1^n) = \mathsf{co}L'(1^n)$; thus, $L' \in \mathsf{P}$, which implies that $L \in \mathsf{E}$. □

**Proposition 6.7** ([KS04]). *If* $\mathsf{Dist}(\mathsf{coNP}) \subseteq \mathsf{Avg}^1_{1-2^{-n}}\mathsf{P}$, *then* pr-$\mathsf{MA} = \mathsf{NP}$.

*Proof.* Let $\mathcal{D} := \{\mathcal{U}_{2n}\}_{n\in\mathbb{N}}$, where $\mathcal{U}_n$ is the uniform distribution on $\{0,1\}^n$. Consider the language $L := \{\, x \in \{0,1\}^* \mid \mathrm{K}^{|x|^2}(x) \geq |x|/4 \,\}$. Since $(L, \mathcal{D}) \in \mathsf{Dist}(\mathsf{coNP}) \subseteq \mathsf{Avg}^1_{1-2^{-n}}\mathsf{P}$, there exists a one-sided-error heuristic algorithm $A$ such that $A$ rejects every $x \in \{0,1\}^* \setminus L$ and $\Pr_{x\sim\{0,1\}^{2n}}[A(x) = L(x)] \geq 2^{-n}$ for every $n \in \mathbb{N}$.

We claim that, for any $n \in \mathbb{N}$, there exists $x \in \{0,1\}^{2n}$ such that $A(x) = 1$. Otherwise, we have $\Pr_{x\sim\{0,1\}^{2n}}[L(x) = 0] \geq 2^{-n}$. However, by the standard counting argument (Fact 3.5), we also have $\Pr_{x\sim\{0,1\}^{2n}}[L(x) = 0] \leq 2^{2n/4-2n} < 2^{-n}$, which is a contradiction.

Now, consider any $\mathsf{MA}$-algorithm $M$. One can nondeterministically derandomize $M$ as follows. Guess $x$ and check if $A(x) = 1$. If $A(x) = 0$, then reject and halt immediately. Otherwise, we use $x$ to construct a pseudorandom generator as in [IW97, KvM02] and derandomize the rest of the $\mathsf{MA}$-algorithm $M$. □

*Proof of Theorem 6.5.* This is immediate from Propositions 6.6 and 6.7 and Lemma 6.2. □

## 6.3 PRG from $\mathrm{MINKT}^{\mathsf{NP}}$

We now show that $\mathrm{GapMINKT}^{\mathsf{NP}} \in \mathsf{P}$ implies the existence of a nearly optimal pseudorandom generator, which implies $\mathsf{P} = \mathsf{BPP}$.

**Theorem 6.8.** *Let $A$ be any $\mathsf{NP}$-hard problem. If $\mathrm{GapMINKT}^A \in \mathsf{P}$, then there exists an explicit pseudorandom generator $G := \{G_n : \{0,1\}^{O(\log n)} \to \{0,1\}^n\}_{n\in\mathbb{N}}$ secure against linear-sized circuits.*

Using the fact that $\mathrm{GapMINKT}^A$ serves as an oracle that tests the hardness of a function (as in [KS04, ABK+06b]), it is not hard to observe the following.

**Proposition 6.9.** *Let $A$ be any oracle. Assume either $\mathrm{GapMINKT}^A \in \mathsf{P}$ or $\{\mathsf{coMINKT}^A\} \times \mathsf{PSamp} \subseteq \mathsf{Avg}^1_{1-2^{-n}}\mathsf{P}$. Then,* pr-$\mathsf{MA} = \mathsf{NP}$.

*Proof.* When $\{\mathsf{coMINKT}^A\} \times \mathsf{PSamp}$ admits a one-sided-error heuristic algorithm, define $L := \{\, x \in \{0,1\}^* \mid \mathrm{K}^{|x|^2,A}(x) \geq |x|/4 \,\}$; then, the remainder of the proof is exactly the same as that of Proposition 6.7.

Similarly, when $\mathrm{GapMINKT}^A \in \mathsf{P}$, one can take an algorithm $M$ for $\mathrm{GapMINKT}^A$ and consider a language $L := \{\, x \in \{0,1\}^* \mid M(x, 1^{|x|^2}, 1^{|x|/4}) = 0 \,\}$. It is clear that any string with low circuit complexity is not in $L$ and there exists some string in $L$, which can be used to derandomize pr-$\mathsf{MA}$. □

*Proof of Theorem 6.8.* Assume that $\mathrm{GapMINKT}^A \in \mathsf{P}$. By Theorem 5.1, we obtain $\mathsf{NE} \subseteq \mathsf{E}^{\mathsf{NP}} \subseteq \mathsf{E}^{\mathrm{GapMINKT}^A} = \mathsf{E}$. Moreover, by Proposition 6.9, we have pr-$\mathsf{MA} = \mathsf{NP}$. Applying Lemma 6.2, we obtain a nearly optimal pseudorandom generator. □

One can use a similar argument, i.e., replacing Theorem 5.1 with Theorem 5.3, to prove the following.

**Theorem 6.10.** *Let $A$ be any $\mathsf{NP}$-oracle. If $\{\mathrm{coMINKT}^A\} \times \mathsf{PSamp} \subseteq \mathsf{Avg}^1_{1-2^{-n}}\mathsf{P}$, then there exists a nearly optimal pseudorandom generator.*

# 7 DistPH-hardness of GapMINKT$^{\mathsf{PH}}$

This section is devoted to proving the DistPH-hardness of $\mathrm{GapMINKT}^{\mathsf{PH}}$. One of the important components of the proof is the $k$-wise direct product generator, which achieves nearly optimal advice complexity.

**Theorem 7.1** ($k$-wise direct product generator [Hir20b, Hir20a])**.** *For any parameters $n, k \in \mathbb{N}$ and $\epsilon > 0$ such that $k \leq 2n$, there exists a "black-box pseudorandom generator construction" $(\mathrm{DP}, A^{(\cdot)}, R^{(\cdot)})$ satisfying the following.*

1. $\mathrm{DP}\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{d+k}$.
2. $A^D\colon \{0,1\}^n \times \{0,1\}^r \to \{0,1\}^a$.
3. $R^D\colon \{0,1\}^a \times \{0,1\}^r \to \{0,1\}^n$ *for any function* $D\colon \{0,1\}^{d+k} \to \{0,1\}$.
4. *The seed length $d$ is at most $O(k \cdot \log(n/\epsilon))$, the advice complexity $a$ is at most $k + O(\log(k/\epsilon))$, the randomness complexity $r$ is at most $\mathsf{poly}(n/\epsilon)$, and $\mathrm{DP}, A^D$, and $R^D$ are computable in time $\mathsf{poly}(n/\epsilon)$.*
5. *For any function $D$ that $\epsilon$-distinguishes the output distribution of $\mathrm{DP}(x, \cdot)$ from the uniform distribution, it holds that*
$$\Pr_w[R^D(A^D(x,w),w) = x] \geq 3/4.$$

We remark that, unlike the notion of BB-HSG defined in Definition 4.1, we allow the advice function $A$ to have oracle access to $D$. This is simply for amplifying the success probability so that later proofs are clearer. This is contrary to Definition 4.1 and the Composition Theorem, in which it was important that $A$ does not have oracle access to $D$.

*Proof Sketch.* The pseudorandom generator construction is defined as

$$\mathrm{DP}(x, \bar{z}) := z_1 \cdots z_k \cdot \widehat{x}(z_1) \cdots \widehat{x}(z_k),$$

where $\widehat{x} := \mathrm{Enc}(x) \in \{0,1\}^{2^\ell}$ is an error-corrected version of $x$ (Lemma 4.11), $\bar{z} = (z_1, \cdots, z_k) \in (\{0,1\}^\ell)^k = \{0,1\}^d$, and $d := \ell k = O(k \log(n/\epsilon))$. The security of DP can be proved by using a standard hybrid argument, as in [NW94].

We note that the success probability of the reconstruction procedure can be amplified without costing the advice complexity significantly. Specifically, one can use the standard hybrid argument

to construct a reconstruction procedure $R_0^D$ and $A_0^D \colon \{0,1\}^n \times \{0,1\}^{r_0} \to \{0,1\}^a$ such that $a = k + O(\log(k/\epsilon))$ and

$$\Pr_{w_0}\left[R_0^D(A_0^D(x,w_0),w_0) = x\right] \geq \frac{\epsilon}{2k}$$

for any function $D$ that $\epsilon$-distinguishes $\mathrm{DP}(x,\text{-})$. Then, the success probability of the reconstruction procedure $(R_0^D, A_0^D)$ can be amplified to $3/4$ by repeating it $m = O(k/\epsilon)$ times and providing the successful index as advice; a formal description follows. Let $m = O(k/\epsilon)$ and define $A^D \colon \{0,1\}^n \times (\{0,1\}^{r_0})^m \to \{0,1\}^{a+\log m}$ such that $A^D(x,(w_1,\cdots,w_m))$ is equal to the pair $(A_0^D(x,w_i),i)$, where $i \in [m]$ is the first index $i$ such that $R_0^D(A_0^D(x,w_i),w_i) = x$. Define $R^D \colon \{0,1\}^{a+\log m} \times (\{0,1\}^{r_0})^m \to \{0,1\}^n$ such that $R^D((\alpha,i),(w_1,\cdots,w_m)) := R_0^D(\alpha,w_i)$. □

As a consequence, given any oracle $D$ that distinguishes $\mathrm{DP}(x,\text{-})$ from the uniform distribution, one can enumerate a list of $\widetilde{O}(2^k)$ strings that contain $x$ by a randomized algorithm, which we call the "Enumeration Lemma". In order to simplify the proof of DistPH-hardness, we will use its derandomized version.

**Corollary 7.2** (Derandomized Enumeration Lemma)**.** *Assume that there exists a nearly optimal pseudorandom generator. Then, there exist a polynomial-time-computable family of functions* $\mathrm{DP} = \{\mathrm{DP}_{n,k} : \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{d+k}\}_{k \leq 2n}$*, where* $d = O(k \log n)$*, and a polynomial-time algorithm* $E$ *that, given* $(1^n, 1^{2^k}, D)$ *as input, where* $D \colon \{0,1\}^{d+k} \to \{0,1\}$ *is a circuit, outputs a list* $\mathcal{L} \subseteq \{0,1\}^n$ *such that* $x \in \mathcal{L}$ *holds for any string* $x \in \{0,1\}^n$ *such that* $D$ $(1/4)$*-distinguishes* $\mathrm{DP}_{n,k}(x,\text{-})$*.*

*Proof.* Let $\mathrm{DP}_{n,k}$ be the $k$-wise direct product generator of Theorem 7.1 with parameter $\epsilon := 1/4$, and let $A^{(\text{-})}, R^{(\text{-})}$ be its advice and reconstruction functions, respectively. The algorithm $E$ takes $1^n$, $1^{2^k}$, and (the description of) a circuit $D \colon \{0,1\}^{d+k} \to \{0,1\}$ and outputs the list $\mathcal{L} := \{\, R^D(\alpha, G_m(z)) \mid \alpha \in \{0,1\}^{k+O(\log k)}, z \in \{0,1\}^{O(\log m)} \,\}$, where $G_m \colon \{0,1\}^{O(\log m)} \to \{0,1\}^m$ is a nearly optimal pseudorandom generator secure against $O(m)$-size circuits. Here, $m$ is chosen later so that it is larger than the randomness complexity $r$, and the reconstruction procedure $R^D(\alpha, G_m(z))$ uses the first $r$ bits of $G_m(z)$ as the source of randomness.

Fix any string $x \in \{0,1\}^n$ such that $D$ $(1/4)$-distinguishes $\mathrm{DP}_{n,k}(x,\text{-})$. Consider the statistical test $T_x \colon \{0,1\}^r \to \{0,1\}$ defined as

$$T_x(w) := 1 \quad \Longleftrightarrow \quad R^D(A^D(x,w),w) = x$$

for any $w \in \{0,1\}^r$. Note that $T_x$ can be computed by a circuit of size $\mathsf{poly}(n, \mathsf{size}(D))$, where $\mathsf{size}(D)$ denotes the size of the circuit $D$. We choose $m$ large enough so that $m \geq \mathsf{size}(T_x)$. By the security of the pseudorandom generator $G_m$, it follows that

$$\Pr_z\left[R^D(A^D(x,G_m(z)),G_m(z)) = x\right] \geq \Pr_w\left[R^D(A^D(x,w),w) = x\right] - o(1) \geq 3/4 - o(1).$$

In particular, there exists some $z \in \{0,1\}^{O(\log m)}$ such that $R^D(A^D(x,G_m(z)),G_m(z)) = x$. Choosing $\alpha := A^D(x,G_m(z))$, we have $x \in \mathcal{L}$ as desired. The running time of $E$ is $\mathsf{poly}(m,n,2^k) = \mathsf{poly}(n, 2^k, \mathsf{size}(D))$. □

In order to prove DistPH-hardness, without loss of generality, we can assume that the distribution is uniform owing to the following.

**Lemma 7.3** (Impagliazzo and Levin [IL90]; see also [BT06a])**.** *Let $A$ be any oracle and $\mathcal{U} = \{\mathcal{U}_n\}_{n\in\mathbb{N}}$ denote the family of the uniform distributions $\mathcal{U}_n$ over $\{0,1\}^n$. If $\mathsf{NP}^A \times \{\mathcal{U}\} \subseteq \mathsf{AvgZPP}$, then $\mathsf{DistNP}^A \subseteq \mathsf{AvgZPP}$.*

*Proof Sketch.* Under the assumption that $\mathsf{NP} \times \{\mathcal{U}\} \subseteq \mathsf{AvgZPP}$, one can invert any distributional one-way function [HILL99, IL89]. Let $(L,\mathcal{D}) \in \mathsf{DistNP}^A$ and $S$ be a polynomial-time sampler for $\mathcal{D}$. One can reduce the task of solving $(L,\mathcal{D})$ to the task of solving $(L\circ S,\mathcal{U})$ via the map $x \mapsto I(x)$, where $I$ is an inverter for the candidate distributional one-way function $S$. □

We also require the fact that $\mathrm{K}(xy) \geq \mathrm{K}(x) + |y| - O(\log(|x|+|y|))$ for most strings $y$, which follows from the symmetry of information.

**Lemma 7.4.** *There exists a universal constant $c_0$ such that, for all large $n, m \in \mathbb{N}$ with $m \leq n$ and any string $x \in \{0,1\}^n$, it holds that*

$$\mathrm{K}(xy) \geq \mathrm{K}(x) + m - c_0 \log n$$

*for at least half of the strings $y \in \{0,1\}^m$.*

*Proof.* The symmetry of information for Kolmogorov complexity [ZL70] implies that

$$\mathrm{K}(xy) \geq \mathrm{K}(x) + \mathrm{K}(y \mid x) - O(\log n).$$

By a simple counting argument (Fact 3.5), we have $\Pr_{y\sim\{0,1\}^m}[\mathrm{K}(y \mid x) \geq m-1] \geq 1/2$. □

Now, we are ready to present the $\mathsf{DistPH}$-hardness result for $\mathrm{GapMINKT}^{\mathsf{PH}}$.

**Reminder of Theorem 1.16.** Let $A$ be any $\Sigma_k^{\mathrm{p}}$-hard problem for some $k \in \mathbb{N}$. If $\mathrm{GapMINKT}^A \in \mathsf{P}$, then $\mathsf{Dist}\Sigma_k^{\mathrm{p}} \subseteq \mathsf{AvgP}$.

*Proof of Theorem 1.16.* Since the theorem is trivial if $k = 0$, we assume $k > 0$, in which case Theorem 6.8 implies the existence of a nearly optimal pseudorandom generator $G = \{G_n : \{0,1\}^{O(\log n)} \to \{0,1\}^n\}_{n\in\mathbb{N}}$. By Proposition 3.4 and Lemma 7.3, it suffices to show that $(L,\mathcal{U}) \in \mathsf{AvgBPP} = \mathsf{AvgP}$ for any $L \in \Sigma_k^{\mathrm{p}}$. However, it is rather complicated to analyze the success probability of $\mathsf{AvgBPP}$; therefore, for simplicity, we present a *deterministic* errorless heuristic algorithm that makes use of the nearly optimal pseudorandom generator.

Let $\tau$ be some polynomial such that $\mathrm{Gap}_\tau\mathrm{MINKT}^A \in \mathsf{P}$, and let $B$ be a polynomial-time algorithm that solves $\mathrm{Gap}_\tau\mathrm{MINKT}^A$.

Fix any $L \in \Sigma_k^{\mathrm{p}}$ and let $V$ be a $\Sigma_k^{\mathrm{p}}$-verifier for $L$; that is, using the notation that $\mathcal{Q}^i := \exists$ if $i$ is odd and $\mathcal{Q}^i := \forall$ otherwise, we have

$$x \in L \iff \exists y_1, \forall y_2, \exists y_3, \ldots, \mathcal{Q}^k y_k, V(x, y_1, y_2, \ldots, y_k) = 1,$$

for any $x \in \{0,1\}^*$, where $y_i \in \{0,1\}^{v(|x|)}$ for some polynomial $v$. Since the oracle $A$ is $\Sigma_k^{\mathrm{p}}$-hard, there exists an algorithm that, given $(x, y_1, \ldots, y_{i-1})$ as input and oracle access to $A$, computes the lexicographically first $y_i$ such that $\mathcal{Q}^{i+1}y_{i+1}, \ldots, \mathcal{Q}^k y_k, V(x, y_1, \ldots, y_k) = i \mod 2$ in time $|x|^{c_V}$, where $c_V$ is some universal constant and $i \in [k]$.

We now describe an errorless heuristic scheme $M$ for $(L,\mathcal{U})$. Let $(x,\delta)$ be the input to $M$, where $x \in \{0,1\}^*$ is a string of length $n \in \mathbb{N}$ and $\delta \in (0,1)$ is an error parameter. The algorithm $M$ is

defined recursively, as in Algorithm 1. Algorithm 1 uses the derandomized enumeration procedure $E$ of Corollary 7.2 associated with the $\kappa$-wise direct product generator $\mathrm{DP}_{v(n),\kappa}$. The basic idea behind Algorithm 1 is to replace the quantifiers $\mathcal{Q}^1 y_1 \in \{0,1\}^{v(n)}, \ldots, \mathcal{Q}^k y_k \in \{0,1\}^{v(n)}$ with the quantifiers over the elements in the list enumerated by the algorithm $E$.

**Algorithm 1** Errorless heuristic scheme $M$

**Input:** a string $x \in \{0,1\}^*$ of length $n$ and error parameter $\delta \in (0,1)$.

1: Let $\kappa := c_\kappa \log(n/\delta)$ and $m := n + kv(n) + O(\kappa \log n)$.
2: Let $t_1 := 2n^{c_V}, t_{i+1} := 2\tau(m, t_i)$.
3: Output $\textsc{Enumerate}(1)$.

4: **function** $\textsc{Enumerate}(i \in \mathbb{N}, y_1, \ldots, y_{i-1} \in \{0,1\}^{v(n)})$
5: **if** $i = k+1$ **then**
6: **return** $V(x, y_1, \ldots, y_k)$.
7: **end if**
8: Define a circuit $D\colon \{0,1\}^{d+\kappa} \to \{0,1\}$ as $D(w) := 1 - B(xy_1 \cdots y_{i-1} w, 1^{t_i}, 1^{n+d+\kappa/4})$.
9: **if** $\textsc{Check}(D) =$ "fail" **then**
10: Output $\perp$ and halt immediately.
11: **end if**
12: Run $E$ on input $(1^{v(n)}, 1^{2^\kappa}, D)$ to obtain a list $\mathcal{L}_i \subseteq \{0,1\}^{v(n)}$.
13: Let $\mathcal{L}'_i := \{\, y_i \in \mathcal{L}_i \mid B(xy_1 \cdots y_i, 1^{t_i}, 1^{n+2i\log t_i}) = 1 \,\}$.
14: **return** $\mathcal{Q}^i y_i \in \mathcal{L}'_i, \textsc{enumerate}(i+1, y_1, \ldots, y_i) = 1$.
15: **end function**

16: **function** $\textsc{Check}(D)$
17: Compute $p := \Pr_{u \sim \{0,1\}^{O(\log s)}}[D(G_s(u)) = 1]$, where $s := \mathsf{size}(D)$.
18: **return** "fail" if $p \le 3/8$; otherwise, **return** "ok".
19: **end function**

First, we claim that the algorithm $M$ outputs some value on most inputs $x \in \{0,1\}^n$.

**Claim 7.5.** $\Pr_{x \sim \{0,1\}^n}[M(x,\delta) = \perp] \le \delta$.

*Proof.* Consider the set $R := \{\, x \in \{0,1\}^n \mid \mathrm{K}^A(x) \ge n - \log 1/\delta \,\}$ of Kolmogorov-random strings. By the standard counting argument (Fact 3.5), it holds that $\Pr_{x \sim \{0,1\}^n}[x \in R] \ge 1 - \delta$. We claim that $M(x,\delta) \ne \perp$ for any $x \in R$. To this end, it suffices to claim that $\Pr_w[D(w) = 1] \ge 1/2$ for any circuit $D$ defined in Line 8. By definition, $D(w) = 1$ if and only if $B(xy_1 \cdots y_{i-1} w, 1^{t_i}, 1^{n+d+\kappa/4}) = 0$. Since $B$ solves $\mathrm{Gap}_\tau \mathrm{MINKT}^A$ and $\mathrm{K}^{\tau(m,t_i),A}(xy_1 \cdots y_{i-1} w) \ge \mathrm{K}^A(xy_1 \cdots y_{i-1} w)$, any $w$ such that $\mathrm{K}^A(xy_1 \cdots y_{i-1} w) > n + d + \kappa/4 + \log \tau(m, t_i)$ is accepted by $D$; note here that the parameter $m$ is chosen so that $m \ge |xy_1 \cdots y_{i-1} w|$. For at least half of the $w \in \{0,1\}^{d+\kappa}$, we have

$$
\begin{aligned}
\mathrm{K}^A(xy_1 \cdots y_{i-1} w) &\ge \mathrm{K}^A(xw) - O(\log n) \\
&\ge \mathrm{K}^A(x) + |w| - O(\log n) && (\text{by Lemma 7.4, for a half of } w) \\
&\ge n - \log 1/\delta + d + \kappa - O(\log n) \\
&> n + d + \kappa/2 \\
&> n + d + \kappa/4 + \log \tau(m, t_i),
\end{aligned}
$$

where the last two inequalities hold for any $i \in [k]$ by choosing sufficiently large constant $c_\kappa$. Therefore,

$$\Pr_w [D(w) = 1] \geq 1/2,$$

from which it follows that $\textsc{Check}(D) =$ "ok". $\diamond$

Consider Line 13, where the algorithm takes the partial list $\mathcal{L}'_i$ of $\mathcal{L}_i$. We observe that the list $\mathcal{L}'_i$ does not contain any string with high Kolmogorov complexity.

**Claim 7.6.** *For any $i \in [k]$ and for the list $\mathcal{L}'_i$ in the algorithm $M$, any string $y_i \in \mathcal{L}'_i$ satisfies*

$$\mathrm{K}^{t_{i+1}/2,A}(xy_1 \cdots y_i) = \mathrm{K}^{\tau(m,t_i),A}(xy_1 \cdots y_i) \leq n + (2i+1)\log t_{i+1},$$

*where $y_1, \cdots, y_{i-1}$ are arbitrary arguments in* Enumerate.

*Proof.* Since $B(xy_1 \cdots y_i, 1^{t_i}, 1^{n+2i\log t_i}) = 1$, the instance $(xy_1 \cdots y_i, 1^{t_i}, 1^{n+2i\log t_i})$ is not a No instance of $\mathrm{Gap}_\tau\mathrm{MINKT}^A$. Therefore, $\mathrm{K}^{\tau(m,t_i),A}(xy_1 \cdots y_i) \leq n + 2i\log t_i + \log\tau(m,t_i) \leq n + (2i+1)\log t_{i+1}$. $\diamond$

Next, we claim that the algorithm $M$ does not err. Suppose that $M$ outputs some value in $\{0,1\}$. This means that, at Line 12 during the execution of $M$, we have $\textsc{Check}(D) \neq$ "fail", and thus $\Pr_w [D(w) = 1] \geq 1/4$. Under this condition, the Derandomized Enumeration Lemma guarantees that the list $\mathcal{L}_i$ contains any $y_i \in \{0,1\}^{v(n)}$ such that $D$ avoids $\mathrm{DP}_{v(n),\kappa}(y_i, \text{-})$.

Let $x \in L$. In this case, we claim that $M$ outputs 1. The correctness is established by choosing $i := k$ in the following claim.

**Claim 7.7.** *Assume $x \in L$. For any $i \in \{0, \cdots, k\}$,*

$$\mathcal{Q}^1 y_1 \in \mathcal{L}'_1, \ldots, \mathcal{Q}^i y_i \in \mathcal{L}'_i, \mathcal{Q}^{i+1} y_{i+1} \in \{0,1\}^{v(n)}, \ldots, \mathcal{Q}^k y_k \in \{0,1\}^{v(n)}, V(x, y_1, \ldots, y_k) = 1. \quad (2)$$

*Here, $\mathcal{L}'_i$ is the list defined in Line 13 and depends on $y_1, \ldots, y_{i-1}$.*

We note that the correctness proof for the case when $x \notin L$ is essentially the same except that the statement of Eq. (2) is negated; thus, we omit the proof and focus on proving Claim 7.7 for the remainder of the proof.

We prove Claim 7.7 by induction on $i = 0, \cdots, k$. When $i = 0$, the claim follows from the assumption that $x \in L$. Assume that the claim is correct up to $i-1$, which means that

$$\mathcal{Q}^1 y_1 \in \mathcal{L}'_1, \ldots, \mathcal{Q}^{i-1} y_{i-1} \in \mathcal{L}'_{i-1}, \mathcal{Q}^i y_i \in \{0,1\}^{v(n)}, \ldots, \mathcal{Q}^k y_k \in \{0,1\}^{v(n)}, V(x, y_1, \ldots, y_k) = 1.$$

Fix any $y_1 \in \mathcal{L}'_1, \ldots, y_{i-1} \in \mathcal{L}'_{i-1}$ such that $\mathcal{Q}^i y_i \in \{0,1\}^{v(n)}, \ldots, \mathcal{Q}^k y_k \in \{0,1\}^{v(n)}, V(x, y_1, \ldots, y_k) = 1$. If $\mathcal{Q}^i = \forall$, the claim is clearly true. Thus, we consider the case when $\mathcal{Q}^i = \exists$. Let $y_i^*$ be the lexicographically first $y_i \in \{0,1\}^{v(n)}$ such that

$$\mathcal{Q}^{i+1} y_{i+1} \in \{0,1\}^{v(n)}, \ldots, \mathcal{Q}^k y_k \in \{0,1\}^{v(n)}, V(x, y_1, \ldots, y_k) = 1. \quad (3)$$

To complete the induction step, it suffices to prove the following.

**Claim 7.8.** *$y_i^* \in \mathcal{L}_i$, and moreover, $y_i^* \in \mathcal{L}'_i$.*

*Proof.* We first prove $y_i^* \in \mathcal{L}_i$. By the Derandomized Enumeration Lemma, it suffices to show that $D$ rejects $\mathrm{DP}_{v(n),\kappa}(y_i^*, z)$ for any $z \in \{0,1\}^d$. The string $xy_1 \dots y_{i-1} \cdot \mathrm{DP}_{v(n),\kappa}(y_i^*, z)$ can be printed with oracle $A$ by first describing $xy_1 \dots y_{i-1}$ using Claim 7.6 and then computing the lexicographically first $y_i^*$ such that Eq. (3) holds. Therefore,

$$\begin{aligned}
&\mathrm{K}^{t_i,A}(xy_1 \dots y_{i-1} \cdot \mathrm{DP}_{v(n),\kappa}(y_i^*, z)) \\
\leq\, &\mathrm{K}^{t_i/2,A}(xy_1 \dots y_{i-1}) + |z| + O(\log n) \\
\leq\, &n + (2i-1)\log t_i + O(\log n) + d \\
\leq\, &n + d + \kappa/4.
\end{aligned}$$

It follows that $B$ accepts $(xy_1 \dots y_{i-1} \cdot \mathrm{DP}_{v(n),\kappa}(y_i^*, z), 1^{t_i}, 1^{n+d+\kappa/4})$; thus, $D$ rejects $\mathrm{DP}_{v(n),\kappa}(y_i^*, z)$ for every $z$.

Having established that $y_i^* \in \mathcal{L}_i$, we claim that $y_i^* \in \mathcal{L}_i'$. This holds because

$$\begin{aligned}
&\mathrm{K}^{t_i,A}(xy_1 \dots y_{i-1} y_i^*) \\
\leq\, &\mathrm{K}^{t_i/2,A}(xy_1 \dots y_{i-1}) + O(\log n) \\
\leq\, &n + (2i-1)\log t_i + O(\log n) \\
\leq\, &n + 2i \log t_i.
\end{aligned}$$

◇
□

# 8 Error-Tolerant Worst-Case-to-Average-Case Reduction

In this section, we present a proof of an error-tolerant non-black-box worst-case-to-average-case reduction. First, it is instructive to recall the proof techniques of [Hir18, Hir20a] that convert any black-box hitting set generator construction to non-black-box worst-case to average-case reductions for MINKT. Based on the new black-box hitting set generator construction of Theorem 4.3, we obtain an improved quality of approximation for a certain range of parameters.

**Theorem 8.1.** *Let $A$ be any oracle. Assume either* $\mathrm{GapMINKT}^A \in \mathsf{P}$ *or* $(\mathsf{coMINKT}^A, \mathcal{D}) \in \mathsf{Avg}^1_{1/4m}\mathsf{P}$ *for any efficiently samplable distribution $\mathcal{D}$. Then, there exists a polynomial-time algorithm that solves the following promise problem* $\Pi = (\Pi_{\mathrm{YES}}, \Pi_{\mathrm{NO}})$.

$$\begin{aligned}
\Pi_{\mathrm{YES}} &:= \{\, (x, 1^t, 1^s) \mid \mathrm{K}^{t,A}(x) \leq s \text{ and } s \leq 2^{\log^{1/3}|x|} \,\}, \\
\Pi_{\mathrm{NO}} &:= \{\, (x, 1^t, 1^s) \mid \mathrm{K}^{\tau(|x|,t)}(x) > \sigma(|x|,t,s) \text{ and } s \leq 2^{\log^{1/3}|x|} \,\},
\end{aligned}$$

*where $\sigma(n,t,s) := 2s + O(\log nt)$ and $\tau$ is some polynomial.*

Following [Hir18], for any constant $c > 0$, we define the family of distributions $\mathcal{D}^c := \{\mathcal{D}^c_m\}_{m\in\mathbb{N}}$ so that $\mathcal{D}^c_m$ is the distribution of $(x, 1^t, 1^s)$ with $t \sim [m]$, $x \sim \{0,1\}^{m-t}$, and $s := |x| - c\log m$. We first prove that computing Kolmogorov complexity on average is easier than approximating Kolmogorov complexity in the worst case.

**Lemma 8.2.** *Let $A$ be any oracle. If* $\mathrm{GapMINKT}^A \in \mathsf{P}$, *then* $(\mathrm{MINKT}^A, \mathcal{D}^c) \in \mathsf{Avg}_{1/4m}\mathsf{P}$ *for some constant $c > 0$.*

*Proof.* Let $M$ be a polynomial-time algorithm that solves $\mathrm{Gap}_\tau\mathrm{MINKT}^A$ for some polynomial $\tau$. Define an algorithm $M'$ that takes $(x, 1^t, 1^s)$ as input so that $M'(x, 1^t, 1^s) := \perp$ if $M(x, 1^t, 1^s) = 1$; otherwise, $M'(x, 1^t, 1^s) := 0$.

We claim that $M'$ is an errorless algorithm for $\mathsf{co}$MINKT. Assume that $M'(x, 1^t, 1^s) = 0$. This happens only if $M(x, 1^t, 1^s) = 0$, which implies that $\mathrm{K}^{t,A}(x) > s$. Therefore, we obtain $(x, 1^t, 1^s) \notin \mathrm{MINKT}$.

We claim that the failure probability of $M'$ over $(x, 1^t, 1^s) \sim \mathcal{D}^c_m$ is at most $1/4m$ for some constant $c$. Observe that $M'$ outputs $\perp$ only if $(x, 1^t, 1^s)$ is not a No instance of $\mathrm{Gap}_\tau\mathrm{MINKT}^A$; thus, it suffices to show that $\mathrm{K}^{\tau(|x|+t),A}(x) \le s + \log\tau(|x|+t)$ with probability at most $1/4m$. Since $|x| + t = m$ and $s = |x| - c\log m$, we have $s + \log\tau(|x| + t) \le |x| - 2\log m$ by choosing $c$ large enough. Therefore,

$$\Pr_{(x,1^t,1^s)\sim\mathcal{D}^c_m}\left[\mathrm{K}^{\tau(m),A}(x) \le s + \log\tau(m)\right] \le \frac{1}{4m}$$

for a sufficiently large $m$. □

*Proof of Theorem 8.1.* By Lemma 8.2 and Fact 3.3, it suffices to show the conclusion of Theorem 8.1 when $(\mathsf{co}\mathrm{MINKT}^A, \mathcal{D}) \in \mathsf{Avg}^1_{1/4k}\mathsf{P}$, where $\mathcal{D} := \mathcal{D}^c$ for some constant $c > 0$. Let $M$ be a one-sided-error heuristic algorithm for $(\mathsf{co}\mathrm{MINKT}^A, \{\mathcal{D}_k\}_{k\in\mathbb{N}})$ with success probability $1 - 1/4k$. Then, for any $k \in \mathbb{N}$ and any $t \in [k]$, the probability that $M$ fails to compute $\mathsf{co}\mathrm{MINKT}^A$ on input $(x, 1^t, 1^{|x|-c\log k})$ over the choice of $x \sim \{0,1\}^{k-t}$ is at most $1/4$. Since the fraction of $x$ such that $(x, 1^t, 1^{|x|-c\log k})$ is a No instance of $\mathsf{co}\mathrm{MINKT}^A$ is at most $1/2$ (by Fact 3.5), we obtain

$$\Pr_{x\sim\{0,1\}^{k-t}}[M(x, 1^t, 1^{|x|-c\log k}) = 1] \ge 3/4 - 1/2 \ge 1/4. \tag{4}$$

We now describe an algorithm $M'$ for solving $\Pi$. Let $(x, 1^t, 1^s) \in \{0,1\}^*$ be an input and $n := |x|$. Take the BB-HSG $H\colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^m$ of Corollary 4.4 for $m := s + c'\log n + c\log(n + t)$, where $c'$ is some large constant chosen later. Since $m \le 2^{\log^{1/3} n} + O(\log n)$ and $d = O(\log n + \log^3 m) = O(\log n)$, there exist some polynomial $\tau_0$ and some large constant $c'$ such that

$$\mathrm{K}^{\tau_0(n,t),A}(H(x, z)) \le \mathrm{K}^{t,A}(x) + c'\log n \tag{5}$$

for any $z \in \{0,1\}^d$. The algorithm $M'$ is defined as $M'(x, 1^t, 1^s) := 1$ if and only if $M(H(x, z), 1^{\tau_0(n,t)}, 1^{m-c\log(n+t)}) = 0$ for any $z \in \{0,1\}^d$.

We claim the correctness of the algorithm. Let $(x, 1^t, 1^s) \in \Pi_{\mathrm{YES}}$. By Eq. (5), for any $z \in \{0,1\}^d$, the instance $(H(x, z), 1^{\tau_0(n,t)}, 1^{m-c\log(n+t)})$ is a No instance of $\mathsf{co}$MINKT and is always rejected by the one-sided-error heuristic algorithm $M$. Thus, $M'$ accepts any $(x, 1^t, 1^s) \in \Pi_{\mathrm{YES}}$.

Conversely, suppose that $M'$ accepts an input $(x, 1^t, 1^s)$. By definition, this means that the function $D\colon \{0,1\}^m \to \{0,1\}$ defined as $D(w) := M(w, 1^{\tau_0(n,t)}, 1^{m-c\log(n+t)})$ rejects $H(x, z)$ for every $z \in \{0,1\}^d$. It follows from Eq. (4) that $D$ $(1/4)$-avoids $H(x, \text{-})$. By the reconstruction property of $H$, there exists a list $\mathcal{L}_D$ of size $2^{2m+O(\log n+\log^3 m)} = 2^{2s+O(\log nt)}$ containing $x$ such that $x$ can be printed in time $\mathsf{poly}(n, t)$ given the index of $x$ as input. Therefore,

$$\mathrm{K}^{\tau(n,t)}(x) \le 2s + O(\log nt)$$

holds for some polynomial $\tau$, which means that $(x, 1^t, 1^s) \notin \Pi_{\mathrm{NO}}$. □

Note that the promise problem defined in Theorem 8.1 is not necessarily disjoint. This motivates us to define the "non-disjoint" promise problem version of $\mathrm{GapMINKT}^A$ as follows.

**Definition 8.3.** *For any oracle $A$ and any function $\tau\colon \mathbb{N}\times\mathbb{N}\to\mathbb{N}$, the promise problem $\mathrm{Gap}_\tau(\mathrm{K}^A \text{ vs } \mathrm{K})$ is defined as the pair*

$$(\mathrm{Gap}_\tau\mathrm{MINKT}^A_{\textsc{Yes}}, \mathrm{Gap}_\tau\mathrm{MINKT}_{\textsc{No}}),$$

*where, for a promise problem $\Pi$, $\Pi_{\textsc{Yes}}$ and $\Pi_{\textsc{No}}$ denote the sets of the* Yes *and* No *instances of $\Pi$, respectively. We say $\mathrm{Gap}(\mathrm{K}^A \text{ vs } \mathrm{K}) \in \mathsf{P}$ if there exists some polynomial $\tau$ such that $\mathrm{Gap}_\tau(\mathrm{K}^A \text{ vs } \mathrm{K}) \in \mathsf{P}$. For a complexity class $\mathfrak{C}$, we say $\mathrm{Gap}(\mathrm{K}^{\mathfrak{C}} \text{ vs } \mathrm{K}) \in \mathsf{P}$ if $\mathrm{Gap}(\mathrm{K}^A \text{ vs } \mathrm{K}) \in \mathsf{P}$ for every $A \in \mathfrak{C}$.*

It is easy to see that $\mathrm{Gap}(\mathrm{K}^A \text{ vs } \mathrm{K})$ is a harder problem than $\mathrm{GapMINKT}^A$.

**Fact 8.4.** *For any oracle $A$, if $\mathrm{Gap}(\mathrm{K}^A \text{ vs } \mathrm{K}) \in \mathsf{P}$, then $\mathrm{GapMINKT}^A \in \mathsf{P}$.*

*Proof.* Since $\mathrm{Gap}_\tau\mathrm{MINKT}^A_{\textsc{No}} \subseteq \mathrm{Gap}_\tau\mathrm{MINKT}_{\textsc{No}}$, the identity reduction maps any No instance of $\mathrm{Gap}_\tau\mathrm{MINKT}^A$ to that of $\mathrm{Gap}_\tau(\mathrm{K}^A \text{ vs } \mathrm{K})$. □

We will first present the error-tolerant worst-case-to-average-case reduction as a reduction to the task of avoiding a hitting set generator. Let us define the sublinear-time computability of a hitting set generator such that it is compatible with the definition of time-bounded Kolmogorov complexity.

**Definition 8.5.** *Let $s,t\colon \mathbb{N}\to\mathbb{N}$. A family of functions $H = \{H_n : \{0,1\}^{s(n)} \to \{0,1\}^n\}_{n\in\mathbb{N}}$ is said to be* computable in time $t(n)$ with oracle $A$ *if there exists an oracle algorithm $M$ such that, for every $n \in \mathbb{N}$ and every seed $z \in \{0,1\}^{s(n)}$, given $i$ as input, random access to each bit of $(n,z)$, and oracle access to $A$, it outputs the $i$th bit of $H_n(z)$ for any $i \in [n]$ in time $t(n)$ (i.e., $M^{(n,z),A}(i) = H_n(z)_i$).*

We observe that a one-sided-error heuristic algorithm for coMINKT is essentially equivalent to avoiding a hitting set generator. While we omit a direct proof of the converse direction of Lemma 8.6, it is implied by the worst-case-to-average-case reduction.

**Lemma 8.6.** *Let $\delta\colon \mathbb{N}\to(0,1), t\colon \mathbb{N}\to\mathbb{N}$ be any functions. Let $A$ be any oracle. Assume that $(\mathsf{coMINKT}^A[t = t(n), s = n - \log(n/\delta(n))], \mathcal{U}) \in \mathsf{Avg}^1_{1-\delta(n)}\mathsf{P}$. Then, no hitting set generator $H = \{H_n : \{0,1\}^{n-\log(n/\delta(n))-O(\log n)} \to \{0,1\}^n\}_{n\in\mathbb{N}}$ computable in time $t(n)$ with oracle $A$ is $(1-o(1))\cdot\delta(n)$-secure against $\mathsf{P}$.*

*Proof.* Assume, for a contradiction, that there exists a hitting set generator

$$H = \{H_n : \{0,1\}^{n-\log(n/\delta(n))-O(\log n)} \to \{0,1\}^n\}_{n\in\mathbb{N}}$$

computable in time $t(n)$ with oracle $A$. By the definition of the computability of $H$ (Definition 8.5), we have

$$\mathrm{K}^{t(n),A}(H_n(z)) \le |z| + O(\log n) < n - \log(n/\delta(n)). \tag{6}$$

Let $L := \mathsf{coMINKT}^A[t = t(n), s = n - \log(n/\delta(n))]$, and let $A$ be a one-sided-error heuristic algorithm for $(L,\mathcal{U}) \in \mathsf{Avg}^1_{1-\delta(n)}\mathsf{P}$. We claim that $H$ is $(1-o(1))\cdot\delta(n)$-avoided by $A$, which will complete the proof. By Eq. (6), $H_n(z)$ is a No instance of $L$; therefore, $A(H_n(z)) = 0$.

On the other hand, consider a string $w \sim \{0,1\}^n$ chosen uniformly at random. By Fact 3.5, $\Pr_w[L(w) = 0] \leq 2\delta(n)/n$ holds. Therefore,

$$\Pr_w[A(w) = 1] \geq \Pr_w[A(w) = L(w)] - \Pr_w[L(w) = 0] \geq \delta(n) - 2\delta(n)/n \geq (1 - o(1)) \cdot \delta(n).$$

□

Now, we present the error-tolerant non-black-box worst-case-to-average-case reduction that reduces $\mathrm{Gap}(\mathrm{K}^A \text{ vs } \mathrm{K})$ to the task of avoiding a hitting set generator.

**Theorem 8.7.** *Let $c, c', \gamma > 0$ be any constants, and let $A$ be any oracle. Assume that, for any family of functions $H = \{H_n : \{0,1\}^{n - c' \log n} \to \{0,1\}^n\}_{n \in \mathbb{N}}$ computable in time $n^\gamma$ with oracle $A$, there exists a polynomial-time algorithm $M$ that $n^{-c}$-avoids $H$. Assume that there also exists an explicit pseudorandom generator $G = \{G_n : \{0,1\}^{O(\log n)} \to \{0,1\}^n\}_{n \in \mathbb{N}}$ secure against linear-sized circuits. Then,* $\mathrm{Gap}(\mathrm{K}^A \text{ vs } \mathrm{K}) \in \mathsf{P}$.

*Proof of Theorem 8.7.* Under the assumption that there exists a nearly optimal pseudorandom generator, we have pr-$\mathsf{BPP} = \mathsf{P}$. Thus, it suffices to show that $\mathrm{Gap}_\tau(\mathrm{K}^A \text{ vs } \mathrm{K}) \in \text{pr-}\mathsf{coRP} \subseteq \mathsf{P}$ for some polynomial $\tau$.

In brief, we take $H$ as the $A$-oracle universal Turing machine running in time $n^\gamma$. More formally, for each $n \in \mathbb{N}$ and each $z \in \{0,1\}^{n - c' \log n}$, define the $i$th bit of $H_n(z) \in \{0,1\}^n$ as the output of $U^{(n,z),A}(i)$ if it halts in $n^\gamma$ steps and 0 otherwise, for each $i \in [n]$. The resulting hitting set generator $H$ is computable in time $\widetilde{O}(n^\gamma)$, where the overhead is for counting time steps. By applying the assumption (for a slightly larger $\gamma$), there exists a polynomial-time machine $M_0$ that $n^{-c}$-avoids $H$. By the construction, any string $x \in \{0,1\}^n$ such that $\mathrm{K}^{n^\gamma, A}(x) \leq n - c' \log n - O(\log n)$ is contained in the image of $H_n$.

We now describe a one-sided-error randomized algorithm $M_1$ for solving $\mathrm{Gap}_\tau(\mathrm{K}^A \text{ vs } \mathrm{K})$. Let $(x, 1^t, 1^s)$ be an input, where $x \in \{0,1\}^*$ is a string of length $n \in \mathbb{N}$ and $t, s \in \mathbb{N}$. Let $k \in \mathbb{N}, \epsilon > 0$ be some parameters chosen later, and take the $k$-wise direct product generator $(\mathrm{DP} \colon \{0,1\}^n \times \{0,1\}^d \to \{0,1\}^{d+k}, A, R)$ of Theorem 7.1. Let $\rho \leq (t + \mathsf{poly}(n))^{1/\gamma}$ be some parameter chosen later. The randomized algorithm $M_1$ picks $z \sim \{0,1\}^d$ and $w \sim \{0,1\}^\rho$, and accepts if and only if $M_0(\mathrm{DP}(x,z) \cdot w) = 0$. Let $N := d + k + \rho$ be the input length of $M_0$.

We claim the correctness of the algorithm $M_1$ below.

**Claim 8.8.**

1. *If $\mathrm{K}^{t,A}(x) \leq s$, then $M_1(x, 1^t, 1^s)$ accepts with probability 1.*
2. *If $\mathrm{K}^{\tau(n,t)}(x) > s + \log \tau(n,t)$, then $M_1(x, 1^t, 1^s)$ rejects with probability at least $N^{-c}/2$.*

Since $M_1$ is a one-sided-error algorithm, the success probability can be amplified by repeating the computation of $M_1$ for independent random coin flips. Therefore, Claim 8.8 implies that $\mathrm{Gap}_\tau(\mathrm{K}^A \text{ vs } \mathrm{K}) \in \text{pr-}\mathsf{coRP}$, and it now remains to prove Claim 8.8.

We first prove Item 1. Assume $\mathrm{K}^{t,A}(x) \leq s$. Fix any $z \in \{0,1\}^d$ and $w \in \{0,1\}^\rho$.

We prove that $\mathrm{K}^{t',A}(\mathrm{DP}(x,z) \cdot w) \leq N - c' \log N - O(\log N)$, where $t' = |\mathrm{DP}(x,z) \cdot w|^\gamma = N^\gamma \geq \rho^\gamma$ for appropriate choices of the parameters $k, \rho$. Each bit of the string $\mathrm{DP}(x,z) \cdot w$ can be described by $z \in \{0,1\}^d, w \in \{0,1\}^\rho$, and the program $P$ of size $\mathrm{K}^{t,A}(x)$ for describing $x$ as follows: Given an index $i \in [d + k + \rho]$, random access to the descriptions $(z, w, P)$, and oracle access to $A$, if

$i \le d+k$, compute the output $x$ of the program $P$ and output the $i$th bit of $\mathrm{DP}(x,z)$; if $i > d+k$, simply output the $(i-d-k)$th bit of $w$. Since DP can be computed in time $\mathsf{poly}(n)$, it follows that

$$\begin{aligned}\mathrm{K}^{t+\mathsf{poly}(n),A}(\mathrm{DP}(x,z)\cdot w) &\le d + \mathrm{K}^{t,A}(x) + \rho + O(\log n)\\ &\le d + s + \rho + O(\log n),\end{aligned}$$

where the additive term $O(\log n)$ comes from a self-delimiting encoding of the tuple $(z,w,P)$. By setting $\rho := (t+\mathsf{poly}(n))^{1/\gamma}$ and $k := s + O(c'\log N)$, we have

$$\mathrm{K}^{t',A}(\mathrm{DP}(x,z)\cdot w) \le d + k + \rho - c'\log N - O(\log N) = N - c'\log N - O(\log N) \tag{7}$$

as desired.

By the definition of $H$, the string $\mathrm{DP}(x,z)\cdot w$ is in the image of $H_N$; thus, it is rejected by $M_0$. Therefore, $M_1$ accepts $(x, 1^t, 1^s)$, which completes the proof of Item 1.

We now prove Item 2 by its contrapositive. Assume that $M_1(x,1^s,1^t)$ rejects with probability less than $N^{-c}/2$. This means that

$$\Pr_{zw\sim\{0,1\}^{d+\rho}}[M_0(\mathrm{DP}(x,z)\cdot w)=1] < \frac{1}{2}\cdot N^{-c}.$$

Take the secure pseudorandom generator $G_m\colon \{0,1\}^{O(\log m)} \to \{0,1\}^{\rho}$,[26] where $m \le \mathsf{poly}(n)$ is chosen large enough so that $m \ge 8N^c$, $m \ge \rho$, and $m$ is larger than the size of a circuit simulating $M_0(\text{-})$ on inputs of length $N$. Since $G_m$ $(N^{-c}/8)$-fools $M_0(\mathrm{DP}(x,z)\cdot\text{-})$ for any $x,z$, it follows that

$$\Pr_{zw_0\sim\{0,1\}^{d+O(\log m)}}[M_0(\mathrm{DP}(x,z)\cdot G_m(w_0))=1] < \frac{3}{8}\cdot N^{-c}.$$

On the other hand, since $M_0$ $N^{-c}$-avoids $H_N$, we also have

$$\Pr_{w'w\sim\{0,1\}^{d+k+\rho}}\left[M_0(w'\cdot w)=1\right] \ge N^{-c}.$$

By using the security of the pseudorandom generator $G_m$, it follows that

$$\Pr_{w'w_0\sim\{0,1\}^{d+k+O(\log m)}}\left[M_0(w'\cdot G_m(w_0))=1\right] \ge \frac{7}{8}\cdot N^{-c}.$$

Therefore,

$$\Pr_{w',w_0}\left[M_0(w'\cdot G_m(w_0))=1\right] - \Pr_{z,w_0}[M_0(\mathrm{DP}(x,z)\cdot G_m(w_0))=1] \ge \frac{1}{2}\cdot N^{-c}.$$

By an averaging argument, there exists $w_0 \in \{0,1\}^{O(\log m)}$ such that

$$\Pr_{w'}\left[M_0(w'\cdot G_m(w_0))=1\right] - \Pr_{z}[M_0(\mathrm{DP}(x,z)\cdot G_m(w_0))=1] \ge \frac{1}{2}\cdot N^{-c}.$$

This means that, for the function $D\colon \{0,1\}^{d+k}\to\{0,1\}$ defined as $D(w') := M_0(w'\cdot G_m(w_0))$, the black-box PRG construction $\mathrm{DP}(x,\text{-})$ is $(N^{-c}/2)$-distinguished by $D$. Choosing $\epsilon := N^{-c}/2$, by the reconstruction property of DP (Theorem 7.1),

$$\Pr_{w'\sim\{0,1\}^r}\left[R^D(A^D(x,w'),w')=x\right] \ge \frac{3}{4}, \tag{8}$$

[26] The output length of $G_m$ is $m$, but we only use the first $\rho$ bits.

where the advice complexity $a$ is at most $k + O(\log(k/\epsilon)) = s + O(\log N)$. Now, we derandomize the random choice of $w'$ of Eq. (8) using the secure pseudorandom generator $G_{m'} \colon \{0,1\}^{O(\log m')} \to \{0,1\}^r$, where $m'$ is chosen later. To this end, we define a statistical test $T \colon \{0,1\}^r \to \{0,1\}$ as $T(w') := 1$ iff $R^D(A^D(x,w'),w') = x$ for each $w' \in \{0,1\}^r$. By inspection, one can observe that the function $T$ can be computed by a circuit of size $\mathsf{poly}(N)$. We choose $m' \geq \mathsf{poly}(N)$ so that $G_{m'}$ $o(1)$-fools $T$. Then, it follows from Eq. (8) that

$$\Pr_{w'_0 \sim \{0,1\}^{O(\log m')}} \left[ R^D(A^D(x, G_{m'}(w'_0)), G_{m'}(w'_0)) = x \right] \geq \frac{3}{4} - o(1).$$

In particular, there exists some seed $w'_0 \in \{0,1\}^{O(\log m')}$ such that $R^D(\alpha, G_{m'}(w'_0)) = x$ for the advice string $\alpha := A^D(x, G_{m'}(w'_0)) \in \{0,1\}^a$.

Now, we present a program $P'$ of size $s + \log \tau(n,t)$ that describes $x$. The program $P'$ takes the advice string $\alpha = A^D(x, G_{m'}(w'_0))$ and the random bits $w_0 \in \{0,1\}^{O(\log m)}$ and $w'_0 \in \{0,1\}^{O(\log m')}$ (as well as parameters such as $N, k$), and it computes and outputs $R^D(\alpha, G_{m'}(w'_0))$ for $D(\text{-}) := M_0(\text{-} \cdot G_m(w_0))$. This program outputs $x$ in polynomial time; thus,

$$\mathrm{K}^{\tau_0(n,t)}(x) \leq a + O(\log N) \leq s + O(\log N)$$

for some polynomial $\tau_0$. By choosing a polynomial $\tau(n,t)$ larger than $\tau_0(n,t)$ and $2^{O(\log N)}$, we obtain

$$\mathrm{K}^{\tau(n,t)}(x) \leq s + \log \tau(n,t).$$

This completes the proof of Claim 8.8. □

**Lemma 8.9.** *Let $\gamma > 0$ be any constant, $\delta \colon \mathbb{N} \to (0,1)$ be any function such that $\delta(n)^{-1} = n^{O(1)}$, and $A$ be any oracle. Assume that $(\mathsf{coMINKT}^A[t = n^\gamma, s = n - \log(n/\delta(n))], \mathcal{U}) \in \mathsf{Avg}^1_{1-\delta(n)}\mathsf{P}$ and that there exists a nearly optimal pseudorandom generator. Then, $\mathrm{Gap}(\mathrm{K}^A \text{ vs } \mathrm{K}) \in \mathsf{P}$.*

*Proof.* This follows by combining Lemma 8.6 and Theorem 8.7. □

We have now arrived at the final error-tolerant non-black-box worst-case-to-average-case reduction.

**Reminder of Theorem 1.15.** Let $c > 0$ be any constant and $A$ be any $\mathsf{NP}$-hard oracle. If $\{\mathsf{coMINKT}^A\} \times \mathsf{PSamp} \subseteq \mathsf{Avg}^1_{1-n^{-c}}\mathsf{P}$, then $\mathrm{Gap}(\mathrm{K}^A \text{ vs } \mathrm{K}) \in \mathsf{P}$.

*Proof.* Using Theorem 6.10, we can construct a nearly optimal pseudorandom generator. The result follows by combining this with Lemma 8.9. □

## 8.1 A Relationship with Hitting Set Generators

In order to include statements on hitting set generators in our equivalence, we prove the following.

**Theorem 8.10.** *Let $A$ be any $\mathsf{NP}$-hard oracle, and let $\gamma > 0$ be any constant. Assume that $\mathsf{P} = \mathsf{ZPP}$, and that, for some constant $c$, no hitting set generator $H = \{H_n : \{0,1\}^{n - c\log n} \to \{0,1\}^n\}_{n \in \mathbb{N}}$ computable in time $n^\gamma$ with oracle $A$ is secure against $\mathsf{P}$. Then, $\mathrm{Gap}(\mathrm{K}^A \text{ vs } \mathrm{K}) \in \mathsf{P}$.*

*Proof.* Under the assumption, we prove that there exists a nearly optimal pseudorandom generator; then, the result immediately follows from Theorem 8.7.

In order to construct a pseudorandom generator, we first prove the following.

**Claim 8.11.** *Under the assumptions,* $\mathsf{E}^{\mathsf{NP}} \subseteq \mathsf{BPE}$.

*Proof.* The idea of the proof is the same as that of Theorem 5.1. We present a randomized reduction from $\mathsf{E}^{\mathsf{NP}}$ to the task of avoiding some hitting set generator. Let $L \in \mathsf{E}^{\mathsf{NP}}$. Fix any input $x \in \{0,1\}^*$ of length $n$. As in Theorem 5.1, one can take a function $f_x \colon \{0,1\}^{O(n)} \to \{0,1\}^*$ computable in $\mathsf{E}^{\mathsf{NP}} \subseteq \mathsf{E}^A$ such that $L$ can be computed in time $2^{O(n)}$ if one can efficiently enumerate a list of $2^{O(n)}$ functions that contains $f_x$. Therefore, our goal is to show that there exists a $2^{O(n)}$-time randomized algorithm that, on input $x$, outputs a list $\mathcal{L}$ that contains $f_x$ with high probability. Let $N := 2^{O(n)}$ be the size of the truth table of $f_x$, and identify $f_x$ with its truth table $f_x \in \{0,1\}^N$.

Take the BB-HSG $H' \colon \{0,1\}^N \times \{0,1\}^d \to \{0,1\}^m$ of Corollary 4.4, where $d = O(\log N + \log^3 m) = O(\log N)$ and $m = O(\log N) = O(n)$ is a parameter chosen later. Choose $N' = 2^{O(n)}$ large enough so that $f_x$ is computable in time $(N')^{\gamma/2}$ with oracle $A$. Consider the hitting set generator $H_{N'} \colon \{0,1\}^{N'-c\log N'} \to \{0,1\}^{N'}$ defined as $H_{N'}(x,z,w) := H'(f_x,z) \cdot w$, where $z \in \{0,1\}^d$ and $w \in \{0,1\}^{N'-m}$. To make it well defined, we must ensure that $n + d + (N' - m) \leq N' - c\log N'$, which is satisfied by choosing sufficiently large $m = O(n)$.

One can observe that $H_{N'}$ is computable in time $(N')^\gamma$ with oracle $A$: given an index $i \in [N']$ and random access to $(x,z,w)$ (as well as oracle access to $A$), if $i' \leq m$, output the $i'$th bit of $H'(f_x,z)$ in time $(N')^{\gamma/2}$; otherwise, output the $(m-i')$th bit of $w$.

By the assumption, there exists a $\mathsf{poly}(N')$-time algorithm $M$ that avoids $H_{N'}$. Using $M$, we claim that $H'$ can also be avoided by the randomized algorithm $M'$ defined as $M'(u) := M(u,w)$ for a randomly chosen $w \sim \{0,1\}^{N'-m}$ and for $u \in \{0,1\}^m$. For any $z \in \{0,1\}^d$, we have $H'(f_x,z) \cdot w = H_{N'}(x,z,w)$; thus, $H'(f_x,z) \cdot w$ is contained in the image of $H_{N'}$, which is rejected by $M'$. On the other hand, $\Pr_{u,M'}[M'(u) = 1] = \Pr_{u,w}[M(u,w) = 1] \geq \frac{1}{4}$. By an averaging argument, $\Pr_w\left[\Pr_u[M(u,w) = 1] \geq \frac{1}{8}\right] \geq \frac{1}{8}$. Thus, $M(\text{-},w)$ (1/8)-avoids $H'$ for at least a (1/8)-fraction of $w$.

By the reconstruction property, with probability at least (1/8) over the choice of $w$, there exists a list $\mathcal{L}_{D(\text{-},w)}$ containing $f_x$ that can be computed in time $2^{O(n)}$. The success probability can be amplified by repeating this and taking the union of the produced lists. ◇

**Claim 8.12.** *Under the assumptions,* $\mathsf{E}^{\mathsf{NP}} \subseteq \mathsf{BPE} = \mathsf{ZPE} = \mathsf{E}$ *and* pr-$\mathsf{MA} = \mathsf{NP}$.

*Proof.* One can easily show that $\mathsf{BPE} = \mathsf{ZPE}$ as follows. Define a function $H_n \colon \{0,1\}^{n^{\gamma/2}} \to \{0,1\}^n$ such that $H_n(C)$ is the truth table of the function computed by a circuit represented by $C$. By avoiding this function, one can test whether a function has high circuit complexity, which enables us to partially derandomize $\mathsf{BPP}$ to $\mathsf{ZPP}$ by picking a random function $f$, testing its hardness, and using it as a hard function for a pseudorandom generator construction of [IW97]. By a padding argument and the assumption that $\mathsf{ZPP} = \mathsf{P}$, we also have $\mathsf{BPE} = \mathsf{ZPE} = \mathsf{E}$. Similarly, one can prove pr-$\mathsf{MA} = \mathsf{NP}$ (in the same way as for Proposition 6.9). ◇

Therefore, we have $\mathsf{E}^{\mathsf{NP}} \subseteq \mathsf{E}$ and pr-$\mathsf{MA} = \mathsf{NP}$, from which one can construct a pseudorandom generator (Lemma 6.2). □

### 8.2 Monotonicity of Meta-Complexity

We prove the monotonicity of meta-complexity:

**Reminder of Theorem 1.13.** Let $A$ and $B$ be oracles such that $A$ is $\mathsf{NP}$-hard and $B \leq_{\mathrm{T}}^{p} A$. Then, $\mathrm{GapMINKT}^{A} \in \mathsf{P}$ implies $\mathrm{GapMINKT}^{B} \in \mathsf{P}$

We need two simple facts. First, we observe that if *approximating* time-bounded Kolmogorov complexity is easy in the worst case, then computing time-bounded Kolmogorov complexity *exactly* is easy on average. (The proof idea is essentially the same with Lemma 8.2.)

**Lemma 8.13.** *Let $A$ be any oracle. If $\mathrm{GapMINKT}^{A} \in \mathsf{P}$, then there exists a function $\delta\colon \mathbb{N} \to (0,1)$ such that $\delta(n)^{-1} = n^{\Theta(1)}$ and*

$$(\mathrm{MINKT}^{A}[t = n, s = n - \log(n/\delta(n))], \mathcal{U}) \in \mathsf{Avg}_{o(1)}\mathsf{P}.$$

*Proof.* By the assumption, there exists a polynomial $\tau$ such that $\mathrm{Gap}_{\tau}\mathrm{MINKT}^{A} \in \mathsf{P}$. Let $M$ be a polynomial-time algorithm that solves $\mathrm{Gap}_{\tau}\mathrm{MINKT}^{A}$. Define $\delta(n) := \tau(n,n)$.

We claim that an algorithm $M'$ defined below is an errorless heuristic algorithm for $L := \mathrm{MINKT}^{A}[t = n, s = n - \log(n/\delta(n))]$. $M'$ takes an input $x \sim \{0,1\}^{n}$, defines $t := n$ and $s := n - \log(n \cdot \tau(n,t)) = n - \log(n/\delta(n))$, and simulates $M$ on input $(x, 1^{t}, 1^{s})$. $M'$ outputs $\perp$ if $M(x, 1^{t}, 1^{s}) = 1$; otherwise, $M'$ outputs 0.

We claim that $M'$ is errorless. If $M'(x) = 0$, we have $M(x, 1^{t}, 1^{s}) = 0$, which implies that $\mathrm{K}^{t,A}(x) > s$; thus, $x \notin L$.

We claim that the error probability of $M'$ is small. $M'$ outputs $\perp$ only if $M(x, 1^{t}, 1^{s}) = 1$, in which case $(x, 1^{t}, 1^{s})$ is not a No instance of $\mathrm{Gap}_{\tau}\mathrm{MINKT}^{A}$; that is, $\mathrm{K}(x) \leq \mathrm{K}^{\tau(n,t)}(x) \leq s + \log \tau(n,t) = n - \log n$. Therefore, $\Pr_{x \sim \{0,1\}^{n}}[M'(x) = \perp] \leq \Pr_{x}[\mathrm{K}(x) \leq n - \log n] = o(1)$. □

Second, we observe that the complexity of $\mathrm{Gap}(\mathrm{K}^{A}$ vs $\mathrm{K})$ is monotone with respect to $A$.

**Lemma 8.14.** *If $B \leq_{\mathrm{T}}^{p} A$ and $\mathrm{Gap}(\mathrm{K}^{A}$ vs $\mathrm{K}) \in \mathsf{P}$, then $\mathrm{Gap}(\mathrm{K}^{B}$ vs $\mathrm{K}) \in \mathsf{P}$.*

*Proof Sketch.* Since $B$ is reducible to $A$, there exists a polynomial $p$ such that $\mathrm{K}^{p(n,t),A}(x) \leq \mathrm{K}^{t,B}(x) + O(1)$ for any $x \in \{0,1\}^{*}$ and $t \in \mathbb{N}$. Given an instance $(x, 1^{t}, 1^{s})$ of $\mathrm{Gap}(\mathrm{K}^{B}$ vs $\mathrm{K})$, one can reduce it to an instance $(x, 1^{p(n,t)}, 1^{s+O(1)})$ of $\mathrm{Gap}(\mathrm{K}^{A}$ vs $\mathrm{K})$. □

*Proof of Theorem 1.13.* Assume $\mathrm{GapMINKT}^{A} \in \mathsf{P}$. Using $\mathsf{NP}$-hardness of $A$ and Theorem 6.8, we obtain a nearly optimal pseudorandom generator. By Lemma 8.13, we obtain an errorless heuristic algorithm for $(\mathrm{MINKT}^{A}[t = n, s = n - \log(n/\delta(n))], \mathcal{U})$, where $\delta(n)^{-1}$ is some polynomial. These results enable us to apply Lemma 8.9 and obtain $\mathrm{Gap}(\mathrm{K}^{A}$ vs $\mathrm{K}) \in \mathsf{P}$. Using Lemma 8.14, we obtain $\mathrm{Gap}(\mathrm{K}^{B}$ vs $\mathrm{K}) \in \mathsf{P}$, which implies $\mathrm{GapMINKT}^{B} \in \mathsf{P}$ by Fact 8.4. □

## A One-Sided-Error versus Errorless

In this section, we present a simple argument proving that the existence of a one-sided-error heuristic scheme for $\mathfrak{C}$ is equivalent to that of an errorless heuristic scheme for $\mathfrak{C}$, where $\mathfrak{C}$ is an arbitrary complexity class closed under taking the complement.

**Proposition A.1.** *Let $\mathfrak{C}$ be an arbitrary complexity class such that $\mathfrak{C} = \mathsf{co}\mathfrak{C}$. Then,* $\mathsf{Dist}\mathfrak{C} \subseteq \mathsf{AvgP}$ *if and only if* $\mathsf{Dist}\mathfrak{C} \subseteq \mathsf{Avg}^1\mathsf{P}$

*Proof.* The "only if" part is obvious. To prove the converse direction, we prove the following.

**Claim A.2.** *Let $(L, \mathcal{D})$ be any distributional problem and $\delta\colon \mathbb{N} \to (0, 1/2)$ be any function. If $(L, \mathcal{D}) \in \mathsf{Avg}^1_\delta\mathsf{P}$ and $(\mathsf{co}L, \mathcal{D}) \in \mathsf{Avg}^1_\delta\mathsf{P}$, then $(L, \mathcal{D}) \in \mathsf{Avg}_{2\delta}\mathsf{P}$.*

*Proof.* Let $M_1$ be a one-sided-error algorithm for $(L, \mathcal{D}) \in \mathsf{Avg}^1_\delta\mathsf{P}$ and $\neg M_0$ be a one-sided-error algorithm for $(\mathsf{co}L, \mathcal{D}) \in \mathsf{Avg}^1_\delta\mathsf{P}$. This means that $M_1$ rejects every No instance and accepts most Yes instances of $L$, and that $M_0$ accepts every Yes instance and rejects most No instances of $L$. We define an algorithm $M$ as $M(x) := 1$ if $M_1(x) = 1$, $M(x) := 0$ if $M_0(x) = 0$, and $M(x) := \bot$ otherwise. It is clear that $M$ is an errorless algorithm for $L$.

We then bound the failure probability of $M$. If $M(x) = \bot$, then we have $M_1(x) = 0$ and $M_0(x) = 1$; since $L(x) = 0$ or $L(x) = 1$, we must have either $M_1(x) = 0$ and $L(x) = 1$ or $M_0(x) = 1$ and $L(x) = 0$. The probabilities that these events occur can be bounded by the failure probabilities $\delta$ of $M_1$ and $M_0$, respectively. Therefore, for any $n \in \mathbb{N}$,

$$\Pr_{x\sim\mathcal{D}_n}[M(x) = \bot] \leq \Pr_{x\sim\mathcal{D}_n}[M_1(x) \neq L(x)] + \Pr_{x\sim\mathcal{D}_n}[M_0(x) \neq L(x)] \leq 2\delta(n).$$

◇

Proposition A.1 is an immediate corollary of the claim above. □

We note that the same argument does not work when the failure probability $\delta$ of a one-sided-error algorithm is at least $1/2$. In fact, it is easy to construct a counterexample of Claim A.2 when $\delta \geq 1/2$, as in the next claim.

**Fact A.3.** *Let $\delta \geq 1/2$ be any constant. There exists a distributional problem $(L,\mathcal{U})$ such that $(L,\mathcal{U}) \in \mathsf{Avg}^1_\delta\mathsf{P}$, $(\mathsf{co}L,\mathcal{U}) \in \mathsf{Avg}^1_\delta\mathsf{P}$, and $(L,\mathcal{U}) \notin \mathsf{AvgP}$.*

*Proof Sketch.* Take a balanced language $L$ such that $(L,\mathcal{U}) \notin \mathsf{AvgP}$; for example, we define $L \cap \{0,1\}^n$ as a uniformly-at-random subset of size $2^{n-1}$ for each $n \in \mathbb{N}$. Then, an algorithm $M_1$ that always outputs 0 is a one-sided-error heuristic algorithm with failure probability $\delta$ for $(L,\mathcal{U})$; thus, $(L,\mathcal{U}) \in \mathsf{Avg}^1_\delta\mathsf{P}$. Similarly, $(\mathsf{co}L,\mathcal{U}) \in \mathsf{Avg}^1_\delta\mathsf{P}$. □

## Acknowledgement

This work is supported by ACT-I, JST. We thank anonymous reviewers for helpful comments.

## References

[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.

[AF09] Luis Filipe Coelho Antunes and Lance Fortnow. Worst-Case Running Times for Average-Case Algorithms. In *Proceedings of the Conference on Computational Complexity (CCC)*, pages 298–303, 2009.

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

[AFvMV06] Luis Antunes, Lance Fortnow, Dieter van Melkebeek, and N. V. Vinodchandran. Computational depth: Concept and applications. *Theor. Comput. Sci.*, 354(3):391–404, 2006.

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

[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.

[BCGL92] Shai Ben-David, Benny Chor, Oded Goldreich, and Michael Luby. On the Theory of Average Case Complexity. *J. Comput. Syst. Sci.*, 44(2):193–219, 1992.

[BFP05] Harry Buhrman, Lance Fortnow, and Aduri Pavan. Some Results on Derandomization. *Theory Comput. Syst.*, 38(2):211–227, 2005.

[BH92] Harry Buhrman and Steven Homer. Superpolynomial Circuits, Almost Sparse Oracles and the Exponential Hierarchy. In *Proceedings of the Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 116–127, 1992.

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

[BS07] Andrej Bogdanov and Muli Safra. Hardness Amplification for Errorless Heuristics. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 418–426, 2007.

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

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

[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.

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

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

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

[GNW11] Oded Goldreich, Noam Nisan, and Avi Wigderson. On Yao's XOR-Lemma. In *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation*, pages 273–301. 2011.

[Gol11] Oded Goldreich. A Sample of Samplers: A Computational Perspective on Sampling. In *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation*, pages 302–332. Springer, 2011.

[GSV18] Aryeh Grinberg, Ronen Shaltiel, and Emanuele Viola. Indistinguishability by Adaptive Procedures with Advice, and Lower Bounds on Hardness Amplification Proofs. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 956–966, 2018.

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

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

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

[Hir20a] Shuichi Hirahara. Non-Disjoint Promise Problems from Meta-Computational View of Pseudorandom Generator Constructions. In *Proceedings of the Computational Complexity Conference (CCC)*, pages 20:1–20:47, 2020.

[Hir20b] Shuichi Hirahara. Unexpected hardness results for Kolmogorov complexity under uniform reductions. In *Proceedings of the Symposium on Theory of Computing (STOC)*, pages 1038–1051, 2020.

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

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

[HVV06] Alexander Healy, Salil P. Vadhan, and Emanuele Viola. Using Nondeterminism to Amplify Hardness. *SIAM J. Comput.*, 35(4):903–931, 2006.

[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.

[HW20] Shuichi Hirahara and Osamu Watanabe. On Nonadaptive Security Reductions of Hitting Set Generators. In *Proceedings of the Approximation, Randomization, and Combinatorial Optimization (APPROX/RANDOM)*, pages 15:1–15:14, 2020.

[IJK09] Russell Impagliazzo, Ragesh Jaiswal, and Valentine Kabanets. Approximate List-Decoding of Direct Product Codes and Uniform Hardness Amplification. *SIAM J. Comput.*, 39(2):564–605, 2009.

[IJKW10] Russell Impagliazzo, Ragesh Jaiswal, Valentine Kabanets, and Avi Wigderson. Uniform Direct Product Theorems: Simplified, Optimized, and Derandomized. *SIAM J. Comput.*, 39(4):1637–1665, 2010.

[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.

[IKW02] Russell Impagliazzo, Valentine Kabanets, and Avi Wigderson. In search of an easy witness: exponential time vs. probabilistic polynomial time. *J. Comput. Syst. Sci.*, 65(4):672–694, 2002.

[IL89] Russell Impagliazzo and Michael Luby. One-way Functions are Essential for Complexity Based Cryptography (Extended Abstract). In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 230–235, 1989.

[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.

[Imp95a] Russell Impagliazzo. A Personal View of Average-Case Complexity. In *Proceedings of the Structure in Complexity Theory Conference*, pages 134–147, 1995.

[Imp95b] Russell Impagliazzo. Hard-Core Distributions for Somewhat Hard Problems. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 538–545, 1995.

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

[ISW06] Russell Impagliazzo, Ronen Shaltiel, and Avi Wigderson. Reducing The Seed Length In The Nisan-Wigderson Generator. *Combinatorica*, 26(6):647–681, 2006.

[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.

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

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

[KS04] Johannes Köbler and Rainer Schuler. Average-case intractability vs. worst-case intractability. *Inf. Comput.*, 190(1):1–17, 2004.

[KvM02] Adam R. Klivans and Dieter van Melkebeek. Graph Nonisomorphism Has Subexponential Size Proofs Unless the Polynomial-Time Hierarchy Collapses. *SIAM J. Comput.*, 31(5):1501–1526, 2002.

[KvMS12] Jeff Kinne, Dieter van Melkebeek, and Ronen Shaltiel. Pseudorandom Generators, Typically-Correct Derandomization, and Circuit Lower Bounds. *Computational Complexity*, 21(1):3–61, 2012.

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

[Lev87] Leonid A. Levin. One-way functions and pseudorandom generators. *Combinatorica*, 7(4):357–363, 1987.

[LTW08] Chi-Jen Lu, Shi-Chun Tsai, and Hsin-Lung Wu. On the Complexity of Hardness Amplification. *IEEE Trans. Information Theory*, 54(10):4575–4586, 2008.

[LV92] Ming Li and Paul M. B. Vitányi. Average Case Complexity Under the Universal Distribution Equals Worst-Case Complexity. *Inf. Process. Lett.*, 42(3):145–149, 1992.

[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.

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

[O'D04] Ryan O'Donnell. Hardness amplification within NP. *J. Comput. Syst. Sci.*, 69(1):68–94, 2004.

[OS17] Igor Carboni Oliveira and Rahul Santhanam. Conspiracies Between Learning Algorithms, Circuit Lower Bounds, and Pseudorandomness. In *Proceedings of the Computational Complexity Conference (CCC)*, pages 18:1–18:49, 2017.

[RRV02] Ran Raz, Omer Reingold, and Salil P. Vadhan. Extracting all the Randomness and Reducing the Error in Trevisan's Extractors. *J. Comput. Syst. Sci.*, 65(1):97–128, 2002.

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

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

[San20] Rahul Santhanam. Pseudorandomness and the Minimum Circuit Size Problem. In *Proceedings of the Innovations in Theoretical Computer Science Conference (ITCS)*, pages 68:1–68:26, 2020.

[SS20] Michael Saks and Rahul Santhanam. Circuit Lower Bounds from NP-Hardness of MCSP Under Turing Reductions. In *Proceedings of the Computational Complexity Conference (CCC)*, pages 26:1–26:13, 2020.

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

[SU05] Ronen Shaltiel and Christopher Umans. Simple extractors for all min-entropies and a new pseudorandom generator. *J. ACM*, 52(2):172–216, 2005.

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

[SV10] Ronen Shaltiel and Emanuele Viola. Hardness Amplification Proofs Require Majority. *SIAM J. Comput.*, 39(7):3122–3154, 2010.

[SY96] Rainer Schuler and Tomoyuki Yamakami. Structural Average Case Complexity. *J. Comput. Syst. Sci.*, 52(2):308–348, 1996.

[Tra84] Boris A. Trakhtenbrot. A Survey of Russian Approaches to Perebor (Brute-Force Searches) Algorithms. *IEEE Annals of the History of Computing*, 6(4):384–400, 1984.

[Tre01] Luca Trevisan. Extractors and pseudorandom generators. *J. ACM*, 48(4):860–879, 2001.

[Tre05] Luca Trevisan. On uniform amplification of hardness in NP. In *Proceedings of the Symposium on Theory of Computing (STOC)*, pages 31–38, 2005.

[TUZ07] Amnon Ta-Shma, Christopher Umans, and David Zuckerman. Lossless Condensers, Unbalanced Expanders, And Extractors. *Combinatorica*, 27(2):213–240, 2007.

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

[Uma09] Christopher Umans. Reconstructive Dispersers and Hitting Set Generators. *Algorithmica*, 55(1):134–156, 2009.

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

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

[VV86] Leslie G. Valiant and Vijay V. Vazirani. NP is as Easy as Detecting Unique Solutions. *Theor. Comput. Sci.*, 47(3):85–93, 1986.

[Wat15] Thomas Watson. Query Complexity in Errorless Hardness Amplification. *Comput. Complex.*, 24(4):823–850, 2015.

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

[ZL70] AK Zvonkin and LA Levin. The complexity of finite objects and the algorithmic concepts of randomness and information. *UMN (Russian Math. Surveys)*, 25(6):83–124, 1970.



ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il